USS User Manual

May 22, 2014

USS User Manual

Edition uss-3.0.0
Published 20140522
Copyright © 2008 Astrium Space Transportation and Rovsing A/S

Contents

1 Installation and Getting Started **3**
- 1.1 Installing the Product . . . 3
  - 1.1.1 Prerequisites . . . 3
  - 1.1.2 Installing the product on Linux and Solaris . . . 3
  - 1.1.3 Installing the product on Windows . . . 4
  - 1.1.4 Integration with CGS . . . 4
  - 1.1.5 Integration with CD-MCS . . . 4
- 1.2 Configuring System Settings . . . 4
  - 1.2.1 Location of SCOE files . . . 5
- 1.3 Getting Started . . . 5
  - 1.3.1 Starting the Editor . . . 5
  - 1.3.2 Starting the executor . . . 6

2 Concepts **9**
- 2.1 Editor and Executor . . . 9
- 2.2 Projects . . . 9
  - 2.2.1 MDB . . . 10
- 2.3 Symbol Libraries . . . 10
- 2.4 Data Quality Indication . . . 10

3 Tutorial **13**
- 3.1 Introduction . . . 13
- 3.2 Installation of USS package . . . 14
  - 3.2.1 Lesson in installing USS package . . . 14
- 3.3 Using the Editor GUI . . . 17
  - 3.3.1 Introduction . . . 17
  - 3.3.2 Docking mechanism of editor panels . . . 17
    - 3.3.2.1 Lesson in docking mechanism of editor panels . . . 17
  - 3.3.3 Tool views of the editor . . . 19
    - 3.3.3.1 Lesson in tool views . . . 19
  - 3.3.4 Editor Print Facility . . . 22
    - 3.3.4.1 Lesson in printing of displays . . . 22
- 3.4 Quick Tutorial . . . 24
  - 3.4.1 Preparation . . . 24
  - 3.4.2 Editor Tutorial . . . 25
    - 3.4.2.1 Creating new ground ops (Satmon like) display . . . 25
      - 3.4.2.1.1 Setting project root folder and MDB configuration . . . 25
      - 3.4.2.1.2 Opening and saving displays from and to MDB . . . 25
      - 3.4.2.1.3 Editing displays . . . 26
      - 3.4.2.1.4 Previewing a display . . . 26
    - 3.4.2.2 Converting GWDU ground displays as batch operation . . . 26
    - 3.4.2.3 Checking GWDU display for consistency . . . 26
  - 3.4.3 Executor Tutorial . . . 26
    - 3.4.3.1 Connecting and opening displays . . . 27
    - 3.4.3.2 Show direct commanding (from FWDU display) . . . 27
    - 3.4.3.3 Show GWDU display . . . 27
    - 3.4.3.4 Show commanding via MCS Tools . . . 28
    - 3.4.3.5 Show imported Satmon display samples . . . 28
    - 3.4.3.6 Show PCS display samples . . . 29
- 3.5 Making a Display . . . 29
  - 3.5.1 Introduction . . . 29
  - 3.5.2 Making display . . . 30
    - 3.5.2.1 Lesson in making a display . . . 30
- 3.6 Import existing displays . . . 34

3.6.1 Lesson in importing a display . . . 34
3.6.2 Lesson in batch importing displays . . . 38
3.7 GUI Elements . . . 39
3.7.1 Lesson in using the GUI elements (View Settings and Preferences) . . . 39
3.8 Changing DQI Styles . . . 45
3.8.1 Lesson in viewing and changing DQI files . . . 45
3.8.2 Lesson in adding a DQI file . . . 46
3.9 Create ASCII displays . . . 47
3.9.1 Lesson in creating ASCII displays . . . 47
3.9.2 Lesson in working with ASCII displays (elements defaults and seperators) . . . 52
3.10 Making a Graph Display . . . 53
3.10.1 Lesson in creating Graph displays . . . 53
3.10.2 Lesson in creating Strip-Graph displays . . . 60
3.11 Create Commanding Display . . . 63
3.11.1 Lesson in creating Commanding displays . . . 63
3.12 Navigation Display . . . 67
3.12.1 Lesson in creating Navigation displays . . . 67
3.13 Add Symbols . . . 72
3.13.1 Lesson in creating Symbols . . . 72
3.14 Create New Symbols . . . 78
3.14.1 Lesson in creating Symbols . . . 78
3.15 Use advanced elements . . . 84
3.15.1 Lesson in creating advanced elements . . . 84
3.15.2 Lesson in creating using fluid elements . . . 89
3.16 Change USS Properties File . . . 91
3.16.1 Lesson in USS properties . . . 91
3.17 Select SCOE Files . . . 93
3.17.1 Lesson in changing the SCOE file . . . 93
3.18 Check Consistency . . . 95
3.18.1 Lesson in checking display consistency . . . 95
3.19 MDB Displays . . . 99
3.19.1 Lesson in using the editor MDB interfacing . . . 99
3.19.2 Lesson in using the editor MDB interfacing for Symbol Libraries . . . 105
3.20 Extended Example: Create A Procedural Display . . . 106
3.20.1 Lesson in creating procedural displays . . . 108

4 Editor **115**
4.1 Introduction . . . 115
4.2 The Editor Workspace . . . 115
4.2.1 Arranging Views and Tab Windows . . . 116
4.2.1.1 Docking mechanism of editor panels . . . 116
4.2.1.2 Basic Concepts . . . 117
4.2.1.3 Drag and Drop Windows . . . 117
4.2.1.4 Minimize and Maximize . . . 118
4.2.1.5 Tab Layout . . . 118
4.2.2 Editor Menus . . . 118
4.2.2.1 Editor Edit Menu Item . . . 118
4.2.2.2 Editor View Menu Item . . . 119
4.2.2.3 Editor Tools Menu Item . . . 119
4.2.2.4 Editor Element Menu Item . . . 119
4.2.2.5 Editor Window Menu Item . . . 120
4.2.2.6 Editor Help Menu Item . . . 120
4.3 Working with Displays . . . 120
4.3.1 Merging Displays . . . 121
4.3.2 Previewing Displays . . . 121
4.3.2.1 Animator . . . 122
4.3.2.2 Slider . . . 122
4.3.2.3 Slider With Manual Specification . . . 122
4.3.2.4 Slider With Values From File . . . 123

4.3.2.5 Manipulating previewed elements . . . 123
4.3.2.6 Preview in Executor . . . 123
4.3.3 Target System and DQI Style . . . 123
4.4 Working with Elements . . . 124
4.4.1 Adding and Deleting Elements . . . 124
4.4.2 Editing Element Properties . . . 125
4.4.3 Selecting Elements . . . 125
4.4.4 Basic Operations . . . 126
4.4.5 Zooming In and Out . . . 127
4.4.6 Aligning and Distributing Elements . . . 127
4.4.7 Using the Grid . . . 127
4.4.8 Grouping Elements . . . 127
4.4.9 Working with Depth . . . 127
4.4.10 Changing the Element Default Values . . . 127
4.5 Using The Symbol Library . . . 128
4.5.1 Pre-Defined Dynamic Symbols . . . 128
4.5.2 Creating New Libraries and Symbols . . . 129
4.5.2.1 Advanced Editing of Symbols . . . 129
4.5.2.2 MDB Version Control of Symbol Libraries . . . 130
4.6 Elements' Advanced Properties . . . 131
4.6.1 The Display . . . 131
4.6.2 Label . . . 131
4.6.3 Data Field . . . 132
4.6.3.1 Field Data Formatting . . . 133
4.6.4 Command Button . . . 134
4.6.5 Command List . . . 139
4.6.6 Navigation Button . . . 139
4.6.7 Graphs . . . 140
4.6.7.1 Line Graphs . . . 140
4.6.7.2 Strip Graphs . . . 143
4.6.7.3 Bar Graphs . . . 143
4.6.8 Arc . . . 144
4.6.9 Polyline and Polygon . . . 144
4.6.10 Linear- and Elliptic- Tickmeter, Thermometer and Tankmeter . . . 144
4.6.10.1 Linear Tick Meter . . . 144
4.6.10.2 Elliptic Tick Meter . . . 145
4.6.10.3 Thermometer . . . 146
4.6.10.4 Tank Meter . . . 146
4.6.11 Pipe, Valve and CheckValve . . . 147
4.6.11.1 Valves . . . 147
4.6.11.2 Pipe . . . 147
4.6.12 Input Field . . . 148
4.6.13 Input ComboBox . . . 148
4.6.14 File Chooser . . . 148
4.6.15 Image . . . 148
4.6.15.1 Symbol State Mapping . . . 149
4.7 Data Sources . . . 149
4.7.1 Data Source Dialog . . . 149
4.7.2 Supported Data Source Types . . . 150
4.7.3 Dynamic Properties . . . 151
4.7.4 Expressions . . . 151
4.7.4.1 Operators . . . 152
4.7.4.2 USS Library Extensions . . . 153
4.7.4.3 Functions . . . 153
4.7.4.4 Selected Computations from PREP Library . . . 157
4.7.4.5 Examples . . . 157
4.8 Mission Database . . . 159
4.8.1 Opening Displays from the MDB . . . 159
4.8.2 Saving Displays in the MDB . . . 159

4.8.3 Adding a New Display to the MDB . . . 159
4.8.4 Forced Open From MDB (revert) . . . 160
4.8.5 Delete in MDB . . . 160
4.9 System Configuration Browser . . . 160
4.10 Working with Projects . . . 161
4.10.1 Synoptic Hierarchy . . . 161
4.10.2 Consistency Check . . . 161
4.10.3 Batch operations . . . 172
4.10.4 Auto Saver . . . 172
4.11 Configuring the Editor . . . 173
4.11.1 System Setting . . . 173
4.11.2 Preferences . . . 173
4.11.3 View Settings . . . 174

5 Executor **175**
5.1 Introduction . . . 175
5.1.1 Configuring User Settings . . . 175
5.1.1.1 Prerequisites . . . 175
5.1.1.2 Configuring User Settings . . . 175
5.1.2 Exiting the Executor . . . 176
5.1.2.1 Prerequisites . . . 176
5.1.2.2 Exiting the Executor . . . 176
5.1.2.3 See also . . . 176
5.2 Monitoring and Control Configuration . . . 176
5.2.1 Connecting to System to be Monitored and Controlled . . . 176
5.2.1.1 Prerequisites . . . 176
5.2.1.2 Connecting to System to be Monitored and Controlled . . . 176
5.2.1.3 See also . . . 177
5.2.2 Disconnecting System to be Monitored and Controlled . . . 177
5.2.2.1 Prerequisites . . . 177
5.2.2.2 Disconnecting System to be Monitored and Controlled . . . 177
5.2.2.3 See also . . . 177
5.2.3 Switching Target for Commands . . . 177
5.2.3.1 Switching Target for Commands . . . 177
5.2.4 Checking Acquisition State . . . 178
5.2.4.1 Prerequisites . . . 178
5.2.4.2 Checking Overall Acquisition State . . . 178
5.2.5 Checking Monitoring State . . . 179
5.2.5.1 Display Status Indicator . . . 179
5.2.5.2 Overall Monitoring Status Indicator . . . 180
5.2.5.2.1 Monitoring status history . . . 181
5.2.5.3 Display Status Overview Window . . . 183
5.2.5.3.1 Configuring the Display Status Overview Window . . . 184
5.2.5.3.1.1 Enable the Display Status Overview Window on Executor startup . . . 184
5.2.5.3.1.2 Set the inital Position of the Display Status Overview Window . . . 184
5.2.5.3.1.3 Set the inital Layout Direction of the Display Status Overview Window . . . 184
5.2.5.3.1.4 Enable the Display Title on the Status Buttons . . . 185
5.2.5.3.1.5 Configuring the Button Colors . . . 185
5.2.5.3.1.6 Configuring the Button Icons . . . 185
5.3 Window Handling . . . 185
5.3.1 Saving Window Layout . . . 185
5.3.1.1 Prerequisites . . . 185
5.3.1.2 Saving Window Layout . . . 186
5.3.1.3 See also . . . 186
5.3.2 Loading Window Layout . . . 186
5.3.2.1 Prerequisites . . . 186

5.3.2.2 Loading Window Layout . . . 186
5.3.2.3 See also . . . 187
5.3.3 Loading Display from File System . . . 187
5.3.3.1 Prerequisites . . . 187
5.3.3.2 Loading Display from File System . . . 187
5.3.4 Reloading Display from File System . . . 188
5.3.4.1 Prerequisites . . . 188
5.3.4.2 Reloading Display from File System . . . 188
5.3.5 Resizing Display Window . . . 188
5.3.5.1 Prerequisites . . . 188
5.3.5.2 Resizing Display Window . . . 188
5.3.6 Resetting Display Window to Default Size . . . 188
5.3.6.1 Prerequisites . . . 188
5.3.6.2 Resetting Display Window to Default Size . . . 188
5.3.7 Showing/Hiding Command Responses . . . 188
5.3.7.1 Prerequisites . . . 188
5.3.7.2 Showing/Hiding Command Responses . . . 188
5.3.8 Navigating Display Hierarchy . . . 189
5.3.8.1 Navigating Display Hierarchy . . . 189
5.3.9 Navigating System Configuration Browser . . . 189
5.3.9.1 System Configuration Browser . . . 189
5.3.9.2 System Configuration: Path View . . . 190
5.3.9.3 System Configuration: Hk View . . . 190
5.3.9.4 System Configuration: Ops View . . . 190
5.3.9.5 System Configuration Browser: Quick Graph Menu . . . 191
5.3.10 Navigating to Home Display . . . 191
5.3.10.1 Prerequisites . . . 191
5.3.10.2 Navigating to Home Display . . . 191
5.3.11 Showing/Hiding the Toolbar . . . 192
5.3.11.1 Prerequisites . . . 192
5.3.11.2 Showing/Hiding the Toolbar . . . 192
5.3.12 Closing Display . . . 192
5.3.12.1 Prerequisites . . . 192
5.3.12.2 Closing Display . . . 192
5.3.12.3 See also . . . 192
5.3.13 Closing All Displays . . . 192
5.3.13.1 Prerequisites . . . 192
5.3.13.2 Closing All Displays . . . 192
5.3.13.3 See also . . . 192
5.3.14 Closing Other Displays . . . 192
5.3.14.1 Prerequisites . . . 192
5.3.14.2 Closing Other Displays . . . 193
5.3.14.3 See also . . . 193
5.3.15 Toggling Tabbed Mode . . . 193
5.3.15.1 Toggling Tabbed Mode . . . 193
5.3.16 Undocking Windows . . . 193
5.3.16.1 Prerequisites . . . 193
5.3.16.2 Undocking Windows . . . 193
5.3.17 Docking Windows . . . 193
5.3.17.1 Prerequisites . . . 193
5.3.17.2 Docking Windows . . . 193
5.3.17.2.1 Docking a Single Window . . . 194
5.3.17.2.2 Docking All Windows . . . 194
5.4 Display Interaction . . . 194
5.4.1 Showing Tooltip for Element . . . 194
5.4.1.1 Showing Tooltip for Element . . . 194
5.4.2 Showing Element Properties . . . 195
5.4.2.1 Show Element Properties . . . 195
5.4.2.2 Showing Expression of Display . . . 195

5.4.3 Showing Display Properties . . . 196
5.4.3.1 Prerequisites . . . 196
5.4.3.2 Show Display Properties . . . 196
5.4.4 Copying Command to Clipboard . . . 197
5.4.4.1 Prerequisites . . . 197
5.4.4.2 Copying Command to Clipboard . . . 197
5.4.5 Copying Parameter Name to Clipboard . . . 197
5.4.5.1 Prerequisites . . . 197
5.4.5.2 Copying Parameter Name to Clipboard . . . 197
5.4.6 Issuing Telecommand via Command Button . . . 197
5.4.6.1 Prerequisites . . . 197
5.4.6.2 Issuing Telecommand via Command Button . . . 197
5.4.6.2.1 1-Step-Commanding . . . 198
5.4.6.2.2 2-Step-Commanding . . . 199
5.4.6.3 See also . . . 200
5.4.7 Issuing Telecommand via Command List . . . 200
5.4.7.1 Prerequisites . . . 200
5.4.7.2 Issuing Telecommand via Command List . . . 200
5.4.8 Finding Displays with Parameter References . . . 201
5.4.8.1 Prerequisites . . . 201
5.4.8.2 Finding Display with Parameter References . . . 201
5.4.8.2.1 Find Parameter with No Display Open . . . 201
5.4.8.2.2 Find Parameter from Open Display . . . 202
5.4.9 Finding Text in Display . . . 202
5.4.9.1 Prerequisites . . . 202
5.4.9.2 Finding Text in Display . . . 202
5.4.10 Showing Parameter Values in a Quick Graph . . . 203
5.4.10.1 Prerequisites . . . 203
5.4.10.2 Showing Parameter Values in a Quick Graph . . . 203
5.5 Quick Graph . . . 204
5.5.1 Introduction . . . 204
5.5.2 Adding and Removing Parameters . . . 204
5.5.3 Configuring the Axis Modes . . . 206
5.5.4 Switching the Curve Style . . . 207
5.5.5 Scrolling and Zooming an Axis . . . 207
5.5.6 Understanding the Buffer and Viewport . . . 208
5.5.7 Managing Time and Value Annotations . . . 208
5.6 Freeze Mode . . . 210
5.6.1 Introduction . . . 210
5.6.2 The Freeze Mode Controls . . . 211
5.7 Help . . . 211
5.7.1 Showing Display Help . . . 211
5.7.1.1 Prerequisites . . . 211
5.7.1.2 Showing Display Help . . . 211
5.7.2 Getting Executor Version Information . . . 212
5.7.2.1 Prerequisites . . . 212
5.7.2.2 Getting Executor Version Information . . . 212
5.8 Miscellaneous . . . 212
5.8.1 Print Preview . . . 212
5.8.1.1 Prerequisites . . . 212
5.8.1.2 Print Preview . . . 212
5.8.2 Printing Display . . . 213
5.8.2.1 Prerequisites . . . 213
5.8.2.2 Printing Display . . . 213
5.8.3 Creating Display Snapshot . . . 213
5.8.3.1 Prerequisites . . . 213
5.8.3.2 Creating Display Snapshot . . . 214
5.8.4 Saving a Copy of Current Display . . . 214
5.8.4.1 Prerequisites . . . 214

5.8.4.2 Saving a Copy of Current Display . . . 214
5.8.5 Configuring Status Display . . . 214
5.8.5.1 Prerequisites . . . 214
5.8.5.2 Configuring Status Display . . . 214
5.8.5.3 See also . . . 215
5.8.6 Configuring Data Quality Indicators . . . 215
5.8.6.1 Prerequisites . . . 215
5.8.6.2 Configuring Data Quality Indicators . . . 215
5.8.6.3 See also . . . 215
5.9 Configuring the Executor . . . 215
5.9.1 Preferences . . . 215
5.9.1.1 Project . . . 216
5.9.1.1.1 Setting Project Folder . . . 216
5.9.1.1.2 Setting Home Display . . . 216
5.9.1.2 Commanding . . . 216
5.9.1.2.1 Enabling Commanding . . . 217
5.9.1.2.2 Enabling Direct Commanding . . . 217
5.9.1.2.3 Choosing Command Target . . . 217
5.9.1.2.4 Confirming Tele Commands . . . 217
5.9.1.3 Tab Appearance . . . 217
5.9.1.3.1 Setting Tab Placement . . . 217
5.9.1.3.2 Setting Tab Policy . . . 217
5.9.1.4 Tooltips . . . 217
5.9.1.4.1 Enabling tooltips . . . 218
5.9.1.5 Snapshots . . . 218
5.9.1.5.1 Saving Snapshots . . . 218
5.9.1.6 Reports . . . 218
5.9.1.6.1 Setting Report Folder . . . 219
5.9.1.7 Import . . . 219
5.9.1.7.1 Opening Import Results Dialog . . . 219
5.9.1.8 Freeze Mode . . . 219
5.9.1.8.1 Enabling Freeze Mode . . . 220
5.9.1.8.2 Setting Major Step Interval . . . 220
5.9.1.8.3 Setting Freeze Mode Duration . . . 220
5.9.1.9 Quick Graph . . . 220
5.9.1.9.1 Set time based limit . . . 221
5.9.1.9.2 Set value based limit . . . 221

6 Application Scenarios **223**
6.1 MCS . . . 223
6.2 LAPAP MKII . . . 225
6.3 COL-TRE . . . 226

7 Importing Foreign Display Formats **227**
7.1 PCS/PREP Displays . . . 227
7.2 PWS/FWDU Displays . . . 228
7.2.1 Extracting FWDU Displays from MDB . . . 228
7.2.2 Generating SCOE XML files . . . 229
7.2.3 Converting the FWDU ASCII definition to USS Format . . . 229
7.2.4 Create HTML report of original PWS/FWDU displays . . . 229
7.3 GWDU Displays . . . 229
7.3.1 GWDU to USS Conversion Details . . . 230
7.3.1.1 Conversion of GWDU Display Objects . . . 230
7.3.1.2 Conversion of GWDU Attributes . . . 232
7.4 Satmon Displays . . . 233
7.4.1 Satmon display file format analysis . . . 233
7.4.1.1 Satmon display XML structure . . . 234
7.4.1.2 Mapping of Satmon display Elements to USS elements . . . 234
7.4.1.3 Alphanumeric Satmon Displays . . . 234

7.4.1.4 Plot Displays . . . 236
7.4.1.5 Procedure Displays . . . 238
7.4.2 USS Satmon importer version 2.12 known issues . . . 238
7.4.2.1 Alphanumeric Displays . . . 238
7.4.2.2 Plots . . . 239
7.4.2.3 Procedure Displays . . . 239
7.4.3 USS Satmon importer version 2.15 known issues . . . 239
7.4.3.1 Alphanumeric Displays . . . 239
7.4.3.2 Plots . . . 239
7.4.3.3 Procedure Displays . . . 239
7.5 Updating Configuration Information in Displays . . . 240

8 Localizing Displays For Different Languages **241**
8.1 Introduction . . . 241
8.2 Translation work-flow . . . 241
8.3 Generating skeletons with the Skeleton Generator . . . 241
8.4 Supported attributes . . . 242
8.5 Format of Entry Keys . . . 242

9 Reference **243**
9.1 Menu references for the executor . . . 243
9.1.1 File Menu . . . 243
9.1.2 Navigate Menu . . . 244
9.1.3 View Menu . . . 244
9.1.4 Options Menu . . . 244
9.1.5 Window Menu . . . 245
9.1.6 Help Menu . . . 245
9.2 Display Versions and Compatibility . . . 245
9.3 XML Display File Format Schema . . . 246
9.4 RGB Colors . . . 258

A Example Configuration in MCS Environment **271**
A.1 Editor Parameters . . . 271
A.2 Executor Parameters . . . 272

B How to Interface with new Target Systems **275**
B.1 Introduction . . . 275
B.2 Development Environment . . . 275
B.2.1 Prerequisites . . . 275
B.2.2 Setup of the Development Environment . . . 275
B.2.3 Compiling the Example Adapter . . . 276
B.2.4 Installing and Using the Example Adapter . . . 276
B.2.4.1 Installing the Example Adapter . . . 276
B.2.4.2 Running USS with the Example Adapter . . . 277
B.2.5 The Example Explained . . . 277
B.2.5.1 Introduction . . . 277
B.2.5.2 ExternalSystemAdapterInterface . . . 278
B.2.5.3 DataProvider . . . 279
B.2.5.3.1 Submitting Data to the Display . . . 281
B.2.5.4 The Telecommand Handler . . . 281

C Frequently Asked Questions (FAQ) **285**

D Glossary **287**

E References **295**
E.1 Reference Documents . . . 295
E.2 Other References . . . 295

Index **297**

List of Figures

1 Installation and Getting Started
1.1 Editor window after start 6
1.2 Executor window after start 7

2 Concepts
2.1 System overview 9
2.2 Symbol library with electrical symbols 10
2.3 Data field with data quality indicators 10
2.4 DQI help table for an MCS display 11
2.5 DQI help table for a PWS display 12

List of Tables

4 Editor

4.1 Target systems and DQI style files . . . 124
4.2 Element Properties with "Default Capability" . . . 128
4.3 Conversions . . . 133
4.4 Flags . . . 133
4.5 Examples . . . 134
4.6 List of input field constraints . . . 137
4.7 Supported Data Source Types and Ranges per Property and Element . . . 151
4.8 Operators . . . 152
4.9 Results of dragging TM/TC from the SCB . . . 160
4.10 USS display consistency checker messages . . . 163

5 Executor

5.1 Indication of Acquisition Status for Status of Data Sources . . . 180
5.2 Overall monitoring status for all open displays . . . 181
5.3 Display status overview monitoring status . . . 183
5.4 Freeze mode controls . . . 211
5.5 Setting limits . . . 221

7 Importing Foreign Display Formats

7.1 GWDU display object conversion to USS object . . . 230
7.2 GWDU attributes conversion to USS properties . . . 232
7.3 Satmon "PageDef" root element attributes . . . 234
7.4 Satmon to USS element mapping . . . 234
7.5 Satmon text styles for USS version 2.12 . . . 235
7.6 Satmon text styles added with USS version 2.15. . . . 235
7.7 FixedAlpha attributes . . . 235
7.8 OutputAlpha attributes . . . 235
7.9 PlotDef attributes . . . 237
7.10 PlotLineDef attributes . . . 237
7.11 LeanProcEntry attributes . . . 238

8 Localizing Displays For Different Languages

8.1 Supported attributes . . . 242

B How to Interface with new Target Systems

B.1 ExternalSystemAdapterInterface Methods and their Call Events . . . 279
B.2 DataProvider Methods and their Call Events . . . 280
B.3 TeleCommandHandler Methods and their Call Events . . . 283

Abstract

This document is the user manual for the Unified Synoptic System (USS). USS is a visual front-end for monitoring and control systems. The USS software allows editing and execution of synoptic (graphical) displays within the Columbus program and CGS. It also provides import capabilities for display formats used within FWDU, GWDU, and NASA PCS.

This manual includes installation and usage instructions for the USS software. Note that this document is also available in PDF format.

Chapter 1

Installation and Getting Started

This chapter describes how to install USS, and where to find more detailed information if any problems should occur during the installation. Section 1.2 describes how to adjust USS to the target environment. As an example the configuration of the home display is shown. Section 1.3 shows how to do the first steps with the editor and executor applications.

1.1 Installing the Product

1.1.1 Prerequisites

To install the USS product in the target environment make sure that:

1. The distribution archive is available. For Linux systems, the archive has the file name `uss-x.y.z-linux-i586.tar.gz` where `x.y.z` is the release version identifier (e.g., 2.9.0). For Windows systems, the archive has the file name `uss-x.y.z-windows-i586.zip`.
2. The target environment fulfills the system requirements as specified in the `INSTALL` file of the distribution archive.

1.1.2 Installing the product on Linux and Solaris

To install USS into the target environment:

1. Unpack the distribution archive into a directory on the target host or central file server. If the target directory is `$basedir`, then the distribution unpacks into a directory `$basedir/uss-x.y.z` where x,y,z are version numbers. Change the working directory to `$basedir/uss-x.y.z`.

> IMPORTANT
>
>
>
> The installation path may not contain white space characters!

2. Read the `README` file for general and latest information.
3. Read the `INSTALL` file for detailed installation instructions.
4. Do one of the following:
   - Change the `$PATH` environment variable settings to include `$basedir/uss-x.y.z/bin` into the search path for executables.
   - Alternatively, create symbolic links to the programs in the `bin` directory to a directory that is already on the search path (e.g., `/usr/local/bin`).

5. Check ownership of files. All files under `$basedir/uss-x.y.z` shall be owned by root. By default, USS does not create or modify files under `$basedir`. Therefore, it is possible to install and use USS on a read-only mounted file system.

1.1.3 Installing the product on Windows

To install USS on Windows:

1. Unpack the distribution archive into a directory on the target host: select the file `uss-x.y.z-windows-i586.zip` in the windows explorer. In the context menu (right-click) choose "Extract All..." and extract the file to a directory of your choice, say, `D:\Programs`. Change the working directory to `D:\Programs\uss-x.y.z`.

> IMPORTANT
>
>
>
> The installation path may not contain white space characters!

2. Read the `README` file for general and latest information.
3. Read the `INSTALL` file for detailed installation instructions.
4. Create desktop links for editor and executor:
   - Right click on `uss-editor.bat`, select "Send To | Desktop (create shortcut)".
   - Right click on `uss-executor.bat`, select "Send To | Desktop (create shortcut)".

1.1.4 Integration with CGS

To integrate USS with CGS 6.3.1 or higher, the CGS Installer should be used. It does automatically unpack the USS archives and changes some properties to meet the CGS needs. USS Editor and Executor are integrated in the top level user interfaces of CGS. Generation of XML SCOE files and export of displays from data base to file system is automatically done by CGS when required. USS displays stored in the mission data base are available through HLCL/UCL commanding and CGS screen setups.

For further information refer to CGS User Manual 6.3.1 , Section 7.3.2.4.13 "USS Displays".

1.1.5 Integration with CD-MCS

The USS executor provides status lights indicating telemetry and link status in the status bar of its main window. The configuration of these lights needs to be adjusted for CD-MCS. This is done by renaming the file `cdmcs-server.properties.xml` to `server.properties.xml` in the `etc` directory of the installed product.

1.2 Configuring System Settings

After installation, the system settings should be configured to adjust USS to the target environment. To configure the system settings, follow these steps:

1. Open the file `$basedir/uss-x.y.z/etc/uss.properties` with a text editor.
2. Read the comments in the configuration file and edit settings where necessary.
3. Save the changed file. The new settings will be used the next time one of the USS applications is started.

For settings in the `uss.properties` file that can also be set via the user interface (e.g. via properties dialog) the following rules apply: Settings made in the `uss.properties` file only define initial defaults. Once they are changed in the UI, the UI settings have preference over the `uss.properties` setting. This means that later changes in the `uss.properties` file may be ignored.

The following explains the precedence of `uss.config` over `uss.properties` by explaining the setting of **uss.cmd.telecommanding** and the various effects on the executor depending on where it is set. Settings in `uss.config` have precedence over `uss.properties`. For example the installation configuration is that the **uss.cmd.telecommanding** is set to false in `uss.config`. Changes to **uss.cmd.telecommanding** in the running executor are also saved to `uss.config`. Changes to that value in `uss.properties` while the executor isn't running has no impact on the executor's preferences, because of the already mentioned precedence of `uss.config` over `uss.properties`.

1.2.1 Location of SCOE files

The location for the SCOE files can be set in `uss.properties` as follows:

1. Edit `$basedir/uss-x.y.z/etc/uss.properties` with a text editor and set the corresponding property `uss.scoe.dir` to the new location of the SCOE files.

Example 1.1 Configuring SCOE files in CGS/MCS

In MCS the SCOE files usually are in `$CGS_HOME/etc/mda/ccu/xml`. The location can be set with:

```
uss.scoe.dir = ${CGS_HOME}/etc/mda/ccu/xml
```

1.3 Getting Started

USS consists of two major applications. The first one is the editor (described in Chapter 4) and the second is the executor (described in Chapter 5). The editor is the application for creating USS displays for later execution in the executor. The editor supports the definition of layout, composition and dynamic properties of synoptic displays. The executor executes the displays which have been authored in the editor.

For an introductory tutorial we suggest to continue with Chapter 3.

1.3.1 Starting the Editor

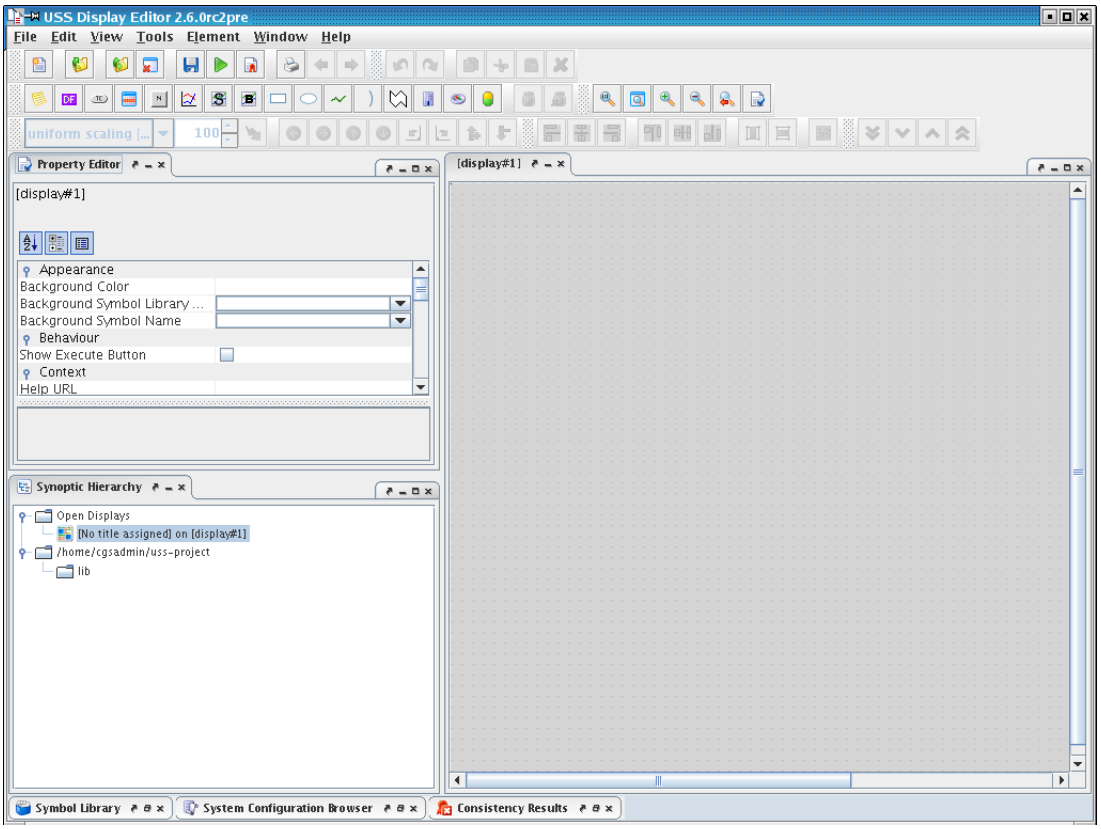
The editor is started via a shell script (batch file on Windows). Open a shell and enter `uss-editor.sh`. The editor will open in a new window (see Figure 1.1).

```
$ uss-editor.sh
```

TIP

In MCS, the editor and executor can be started from the task selector menu.

Figure 1.1 Editor window after start



1.3.2 Starting the executor

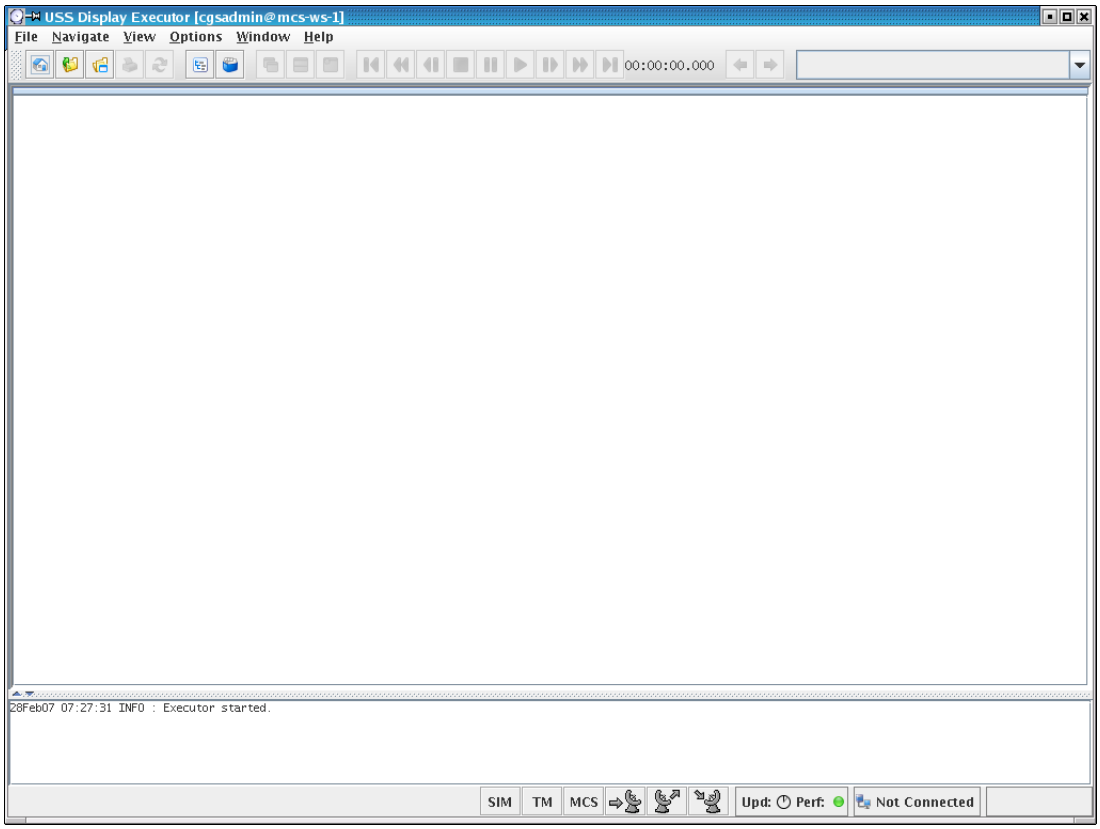
The executor is started via a shell script (batch file on Windows). Open a shell and enter `uss-executor.sh`. The executor will open in a new window (see Figure 1.2).

```
$ uss-executor.sh
```

TIP

In MCS, the editor and executor can be started from the task selector menu.

Figure 1.2 Executor window after start



Editor and executor offer command line options that are primarily useful for integration in special environments. Use the `--help` option for getting help on these.

Chapter 2

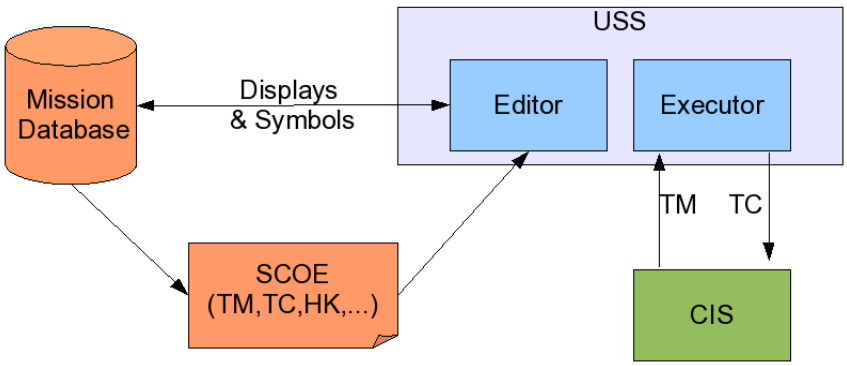
Concepts

This chapter describes the main concepts necessary to understand how USS works.

2.1 Editor and Executor

USS has two main applications: editor and executor. Operational displays are prepared and maintained with the editor. The executor loads and executes displays at mission time. Figure 2.1 shows a high level system overview of USS together with its main interfaces. The editor interfaces with the mission database (MDB) for storing and loading displays and symbols. Note that this is an optional interface. The editor can also directly edit displays in the file system. The second main interface of the editor is the XML SCOE file that represents a snapshot of relevant MDB data including telemetry parameters, telecommands, and house keeping parameters. The SCOE file basically provides a cache for fast access to MDB data.

Figure 2.1 System overview



The executor interfaces with the CGS Corba Interface Server (CIS). This interface is used for acquisition of real-time telemetry and for sending commands.

2.2 Projects

The number of displays usually depends on operational needs. It may range from only a dozen displays to several hundreds or even thousands of displays. Designing and organizing the display structure is the task of the display author. In many cases, the display structure mirrors the hierarchical structure of the space system. A set of displays that belong together to fulfill an operational goal is called a *project* in USS.

Every display is stored in a single file. A project then is nothing more than a set of files on a computer. Authors are free to organize their displays in a hierarchy. The hierarchy is a directory structure containing the displays. This is very similar to a web site storing its documents in a hierarchy tree.

There is exactly one project active in the editor or executor at any time. The project folder is defined the application preferences. The default value is `$HOME/uss-project`. Note that editor and executor can have different project folders but for most users it is recommended to use the same folder in both applications.

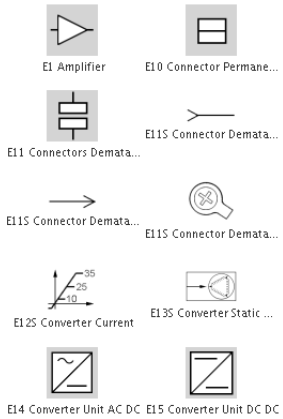
2.2.1 MDB

All USS data can be easily managed in the file system. When working with the MDB things can get a bit more complicated. Different MDB configurations can be used in parallel. This is achieved by having an extra directory below the project folder that is named after the MDB configuration identifier (CDU/CCU internal version identifier). In the editor, this information is determined by the preference settings for the MDB. In the executor, this information is determined by the CIS after a successful connect.

2.3 Symbol Libraries

Symbols are two-dimensional graphical elements showing images or icons. Symbol libraries are groups of symbols organized by topic. For example, the symbol library 'Electrical' contains symbols such as 'Amplifier', 'Connector' or 'Converter' (see Figure 2.2).

Figure 2.2 Symbol library with electrical symbols



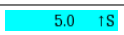
USS comes with a set of predefined symbol libraries. These libraries are called *system libraries*. They contain many of the symbols defined in Appendix C of the International Space Station's Display and Graphics Commonality Standard (DGCS) including: 3D Graphics, Communications, Computers, Electrical, Events, Fluid, General, Groups, Logic, Mechanical, Misc, Modes, PCS, Payloads, Radiation Monitoring, Robotics, Sensors, Station Modules, and Tasks.

Users can define their own libraries in *user libraries*. There is no limit in the number of user symbols. Libraries can be created and symbols can be added and removed from a user library with the USS editor. Symbol libraries can be placed anywhere below a project folder.

2.4 Data Quality Indication

One of the main tasks of the USS executor is the visualization of data. This involves two aspects: (1) the data values must be shown, and (2) the quality of the shown data must be clear. The quality gives an indication about the availability and reliability of the data values and their meaning in the operational context. Most output elements in USS support so-called Data Quality Indicators (DQI) according to the DGCS [DGCS] standard. For example, Figure 2.3 shows a data field with data quality indicators. The cyan background color and the letter 'S' at the end indicate that the parameter value has gone static. At the same time, the arrow before the 'S' shows that the last value had an upper-limit violation.

Figure 2.3 Data field with data quality indicators



Data quality is indicated by foreground and background color of a data field and by additional flags that are shown next to the parameter value. Input for data quality are the acquisition status, monitoring status, and delta monitoring status of a parameter. Data quality indicators can be configured for different environments by editing an XML configuration file. Since the configuration can become quite complex due to the number of possible combinations, the executor has a function to display all settings for a selected display as a help page. Figure 2.4 shows part of the online help generated for an MCS display (by the menu command Help → Data Quality Indicators for Display). This is helpful in two scenarios:

- For administrators defining DQI configurations.
- For operators wanting to see the meaning of DQI output such as colors and flags in a running display.

Figure 2.4 DQI help table for an MCS display

MCS Data Quality Indicators (DQI)

	DQI Symbol Example	Status Acquisition	Monitoring	DeltaMonitoring	
1	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DELTA_DANGER_LIMIT_VIOLATION	ACQU
2	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DELTA_NOMINAL_LIMIT_VIOLATION	ACQU
3	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DISABLED	ACQU
4	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	IN_LIMITS	ACQU
5	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	UNDEFINED	ACQU
6	5.0 ↑	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	UNKNOWN	ACQU
7	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DELTA_DANGER_LIMIT_VIOLATION	ACQU
8	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DELTA_NOMINAL_LIMIT_VIOLATION	ACQU
9	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DISABLED	ACQU
10	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	IN_LIMITS	ACQU
11	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	UNDEFINED	ACQU
12	5.0 ↓	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	UNKNOWN	ACQU
13	5.0 ↕	ACQUIRED	DISABLED	DELTA_DANGER_LIMIT_VIOLATION	ACQU
14	5.0 ↕	ACQUIRED	DISABLED	DELTA_NOMINAL_LIMIT_VIOLATION	ACQU
15	5.0	ACQUIRED	DISABLED	DISABLED	ACQU
16	5.0	ACQUIRED	DISABLED	IN_LIMITS	ACQU
17	5.0	ACQUIRED	DISABLED	UNDEFINED	ACQU
18	5.0	ACQUIRED	DISABLED	UNKNOWN	ACQU
19	5.0	ACQUIRED	IN_LIMITS	DELTA_DANGER_LIMIT_VIOLATION	ACQU
20	5.0	ACQUIRED	IN_LIMITS	DELTA_NOMINAL_LIMIT_VIOLATION	ACQU

DQI settings are defined in style files. These are XML configuration files and can be changed with any editor. For example, the file `etc/mcs_dqistyle.xml` is the configuration for displays in the MCS environment. The file `etc/pws_dqistyle.xml` defines the DQI styles for the PWS displays (onboard Columbus). If you want to define your own DQI styles, it is helpful to understand how USS looks for an applicable style when the value of a data field needs to be updated. The process is as follows:

1. Determine the target of the current display (say 'MCS'). The target of a display is a simple string that can be defined in the editor. It defines the target environment where the display is designed to run. Current targets are PWS, PCS, MCS, and CGS. It is assumed that all displays with the same target should have the same look and feel.

2. Search and read the DQI style file for the target (say `mcs_dqistyle.xml`). A 'target-to-style-file' mapping is defined in `etc/uss.properties`.

3. Look-up the DQI entry for the combination of acquisition status, monitoring status, and delta monitoring status of the current parameter value.

4. Render the data field with the colors and flags found in the previous step.

As a last example, Figure 2.5 shows part of the DQI settings for a PWS display. Compare this with Figure 2.4 and you can see that the same parameter status values can lead to very different output. This flexibility allows to use USS in different operational environments with different demands on visualization and user feedback.

Figure 2.5 DQI help table for a PWS display

PWS Data Quality Indicators (DQI)

	DQI Symbol Example	Status Acquisition	Monitoring	DeltaMonitoring	Desc
1	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DELTA_DANGER_LIMIT_VIOLATION	
2	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DELTA_NOMINAL_LIMIT_VIOLATION	
3	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	DISABLED	
4	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	IN_LIMITS	
5	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	UNDEFINED	
6	5.0	ACQUIRED	DANGER_HIGH_LIMIT_VIOLATION	UNKNOWN	
7	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DELTA_DANGER_LIMIT_VIOLATION	
8	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DELTA_NOMINAL_LIMIT_VIOLATION	
9	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	DISABLED	
10	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	IN_LIMITS	
11	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	UNDEFINED	
12	5.0	ACQUIRED	DANGER_LOW_LIMIT_VIOLATION	UNKNOWN	
13	5.0	ACQUIRED	DISABLED	DELTA_DANGER_LIMIT_VIOLATION	
14	5.0	ACQUIRED	DISABLED	DELTA_NOMINAL_LIMIT_VIOLATION	
15	5.0	ACQUIRED	DISABLED	DISABLED	
16	5.0	ACQUIRED	DISABLED	IN_LIMITS	
17	5.0	ACQUIRED	DISABLED	UNDEFINED	
18	5.0	ACQUIRED	DISABLED	UNKNOWN	
19	5.0	ACQUIRED	IN_LIMITS	DELTA_DANGER_LIMIT_VIOLATION	
20	5.0	ACQUIRED	IN_LIMITS	DELTA_NOMINAL_LIMIT_VIOLATION	

Chapter 3

Tutorial

3.1 Introduction

This tutorial is put together of a series of lessons, going through some of the basic and advanced editing of the USS editor, as well as some advanced topics covering editing external to the USS Editor.

Lessons in creating command elements (9) and navigation elements (10) are made as a continuous extended lessons, but can with slight modification be used separately.

Lesson topics:

1. Installation of USS
2. Usage of the USS Editor
3. Creation of displays
4. Import of display of non-USS format
5. Major GUI elements
6. How to Edit DQI style files
7. Creation of ASCII displays
8. Making a graph display
9. Creation of displays with commands
10. Creation of displays with navigation
11. Adding symbols to displays
12. Creation of new symbols and symbol libraries
13. Creation of displays with advanced elements
14. Changing the USS property file
15. Changing the USS Editor SCOE file
16. Checking Consistency of Displays
17. MDB display actions
18. Quick tutorial
19. An extended example: The creation of a procedural display (a task specific display visually resembling an ODF procedure)

3.2 Installation of USS package

For Prerequisites for installation and other information on installation of USS, see Chapter 1.

USS consists of 3 installation packages for different platforms: GNU/Linux, Solaris and Windows.

A USS package contains among others:

- USS main applications: Editor and Executor
- Java Virtual Machine Environent (Java JRE) version supported by USS
- Execution scripts for platforms
- Example displays
- SCOE files
- Default Symbol libraries

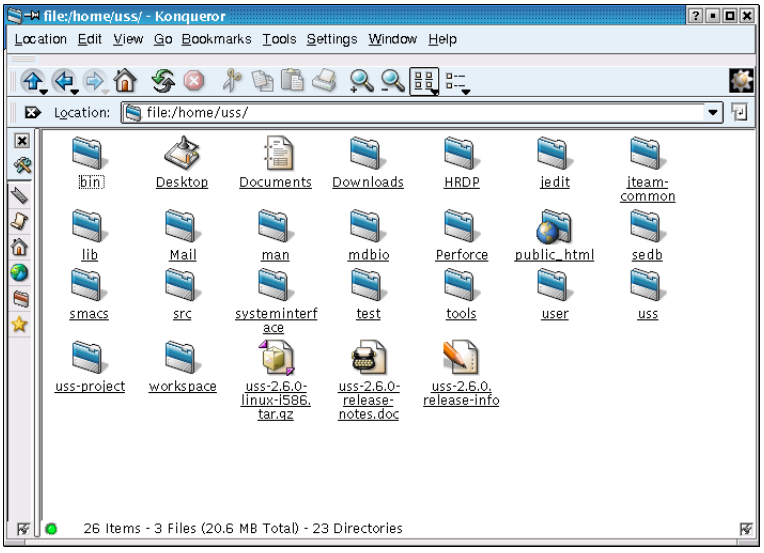
3.2.1 Lesson in installing USS package

Assumption: The archive has the file name `uss-2.6.0-linux-i586.tar.gz` is used with this tutorial in home directory.

1. On the desktop find the Konquerer icon in the task bar, and click on it to start the file-browser.

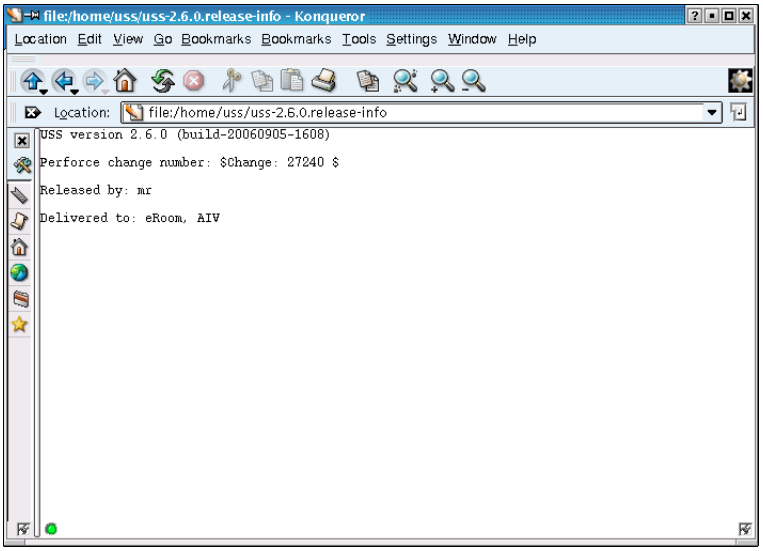
Konquereris normally in the quick start icon bar

2. Konquerer normally opens in the home folder of the current user, click the release info file, to read the release info.



Konquerer showing the content of the user home folder

3. Konquerers embedded text viewer, shows release information. Click the back-button in konquerer to go back to the home folder.

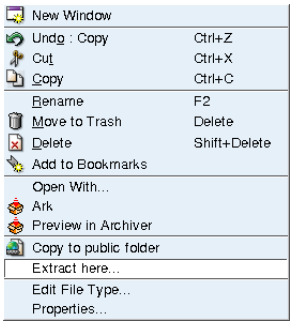

file:/home/uss/uss-2.6.0.release-info - Konqueror
Location Edit View Go Bookmarks Bookmarks Tools Settings Window Help
Location: file:/home/uss/uss-2.6.0.release-info
USS version 2.6.0 (build-20060905-1608)
Perforce change number: $Change: 27240 $
Released by: mr
Delivered to: eRoom, AIV

Release information shown in Konquerer embedded viewer

4. Find the USS package, compressed folder.

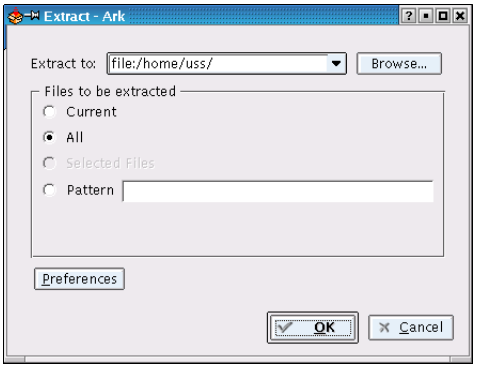
For Linux USS is delivered as a gzip compressed, tar archive

5. Right-click the compressed archive and select 'Extract here' form popup menu.



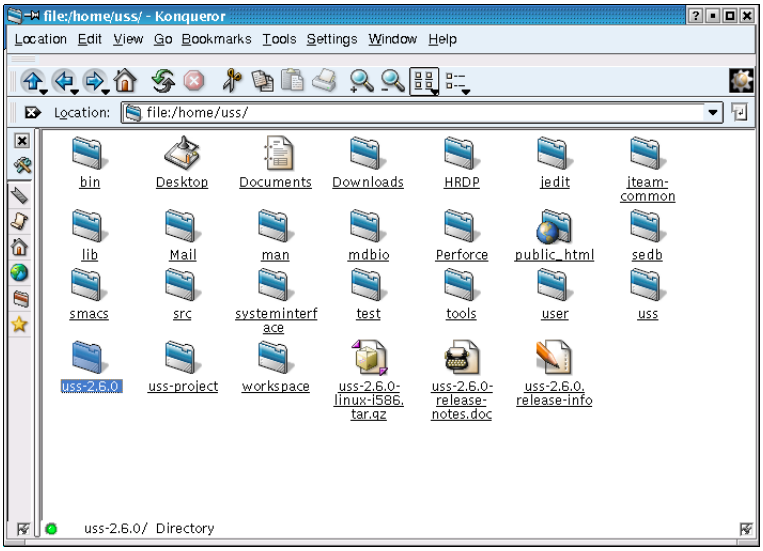
Konquerer popup menu for USS compressed archive

6. Konquerer open default compression/decompression tool, normally Ark. Click OK to decompress in the home folder.



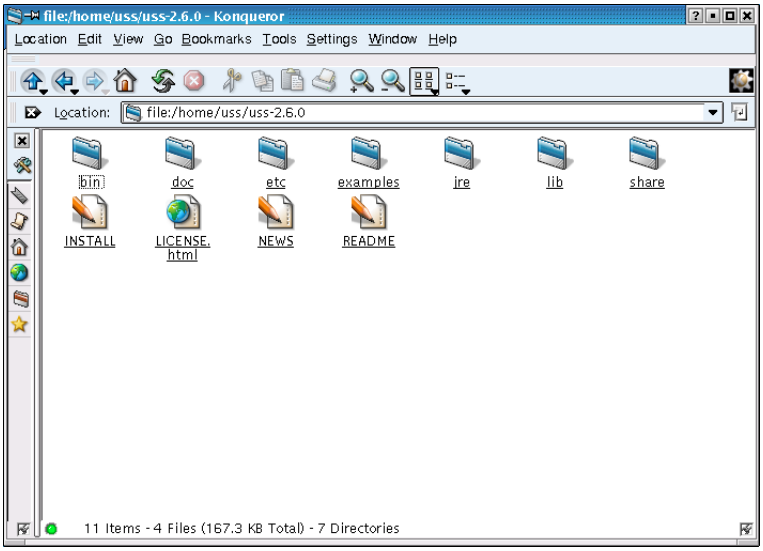
Ark compression tool

7. The decompression of the USS archive generates a USS installation folder.



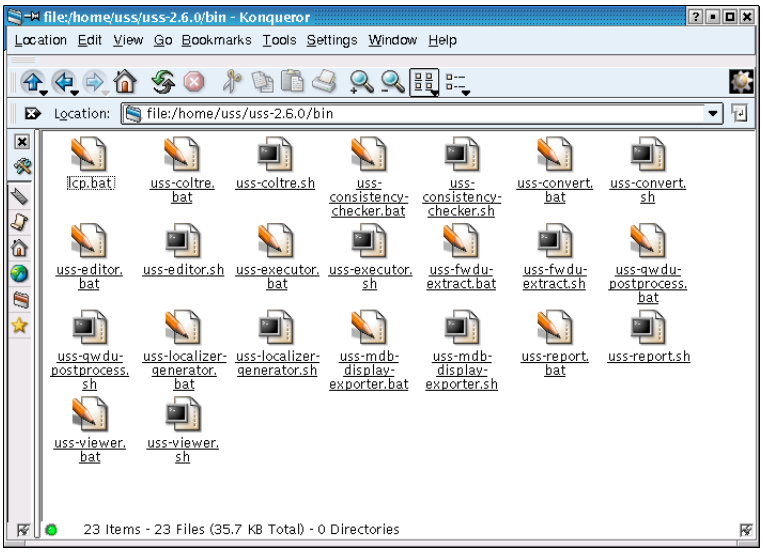
Selected USS installation folder

8. Click to open USS installation folder.



Konquerer showing the contents of USS installation folder

9. Click the bin-folder inside the USS-folder.



Konquerer showing the contents of the uss-bin-folder

10. Find the uss-editor.sh file in the bin-folder.

Selected uss-editor.sh file

11. All shell scripts, i.e. files ending/with extension 'sh' are launch files for the USS applications for UNIX. Likewise all the batch-scripts, i.e. files ending 'bat' are launch files for the USS applications for MS Windows. Click the uss-editor.sh file, and the USS Editor starts.

Start splash picture of the USS Editor

3.3 Using the Editor GUI

3.3.1 Introduction

This lesson will introduce the usage of the basic user interface of the USS Editor. You shall work with the following subjects:

1. Docking mechanism of editor panels
2. Tool views of the editor
3. Printing of displays

3.3.2 Docking mechanism of editor panels

The USS Editor features a docking framework for tool- and display views. By default views are opened as frames inside the main editor application. Small docking control icons allows you to:

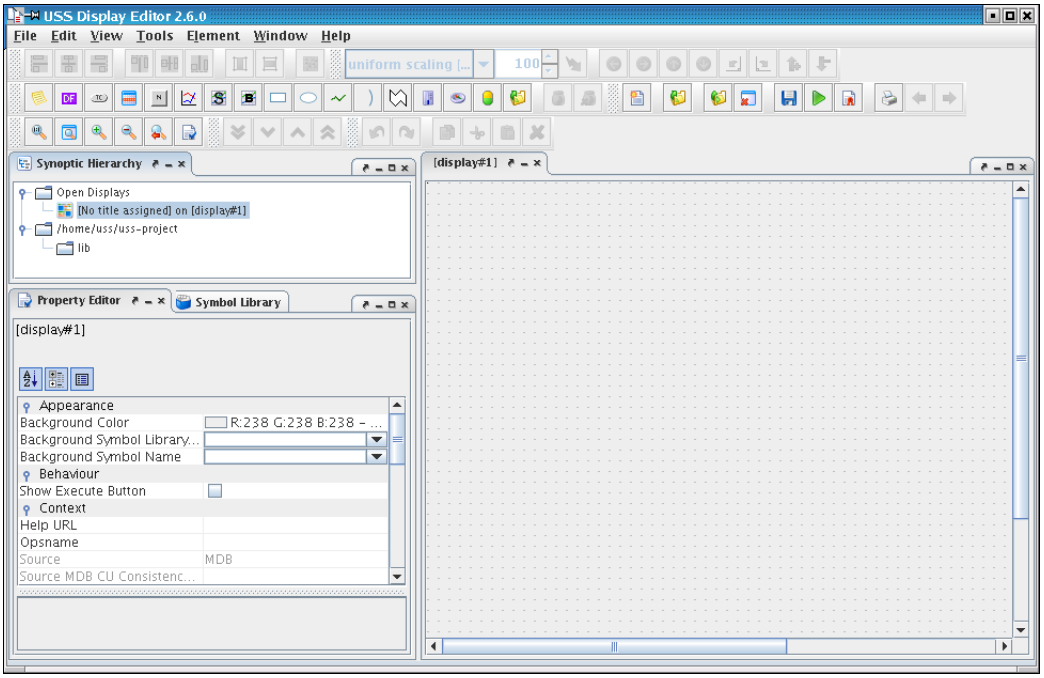
1. Undock/Minimize/Close - when view is docked, i.e.. inside main window

2. Dock/Close - when view is undocked, i.e. appears in a separate window

3. Undock/Minimize/Maximize/Close - when views are collected in tabs (multiple views in tabs)

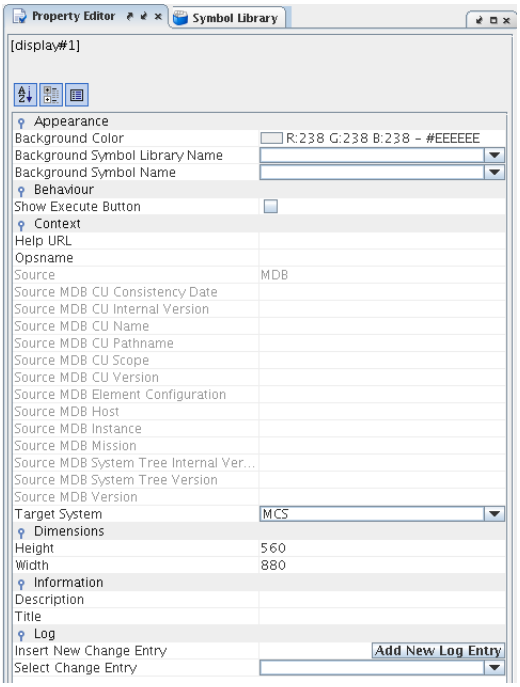
3.3.2.1 Lesson in docking mechanism of editor panels

1. Start by opening the editor via installed icon.



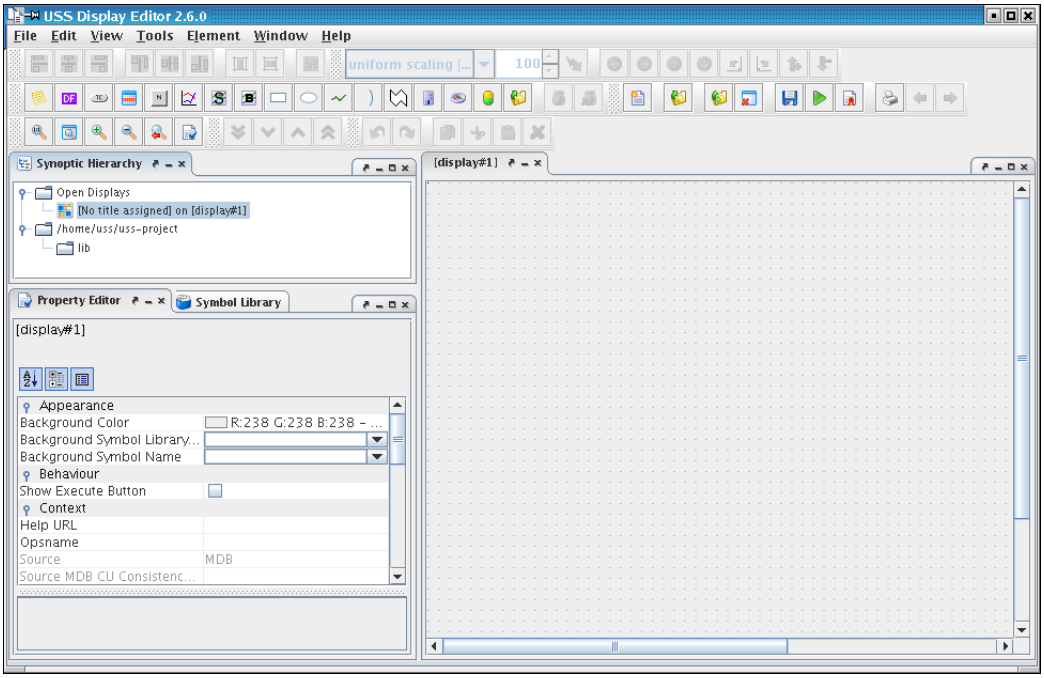
Editor started with default empty display created, three tool views open: Synoptic Hierarchy, Property Editor and Symbol Library. This is an example of start-up layout, the editor saved basic layout and which tools are open from previous editor closing.

2. Undock the view called Property Editor by clicking with left-mouse-button on the undock icon for the Property Editor View. The Property Editor will undock and still function as part of the editor application.



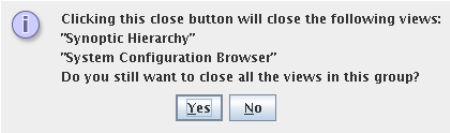
The Property Editor is undocked and can be moved around independently of the main application, on computers with multiple screens, the view can be moved to another screen to better take advanced of the setup.

3. Re-dock the Property Editor by clicking with left-mouse-button on the dock icon in the undocked view. The Property Editor will dock again.



The Property Editor is docked again and can be moved around inside the main application. The view will when moved (click and hold left-mouse-button) dock in different position or on top of other views.

4. Close Property Editor and Symbol Library views by clicking closing their common tab-view.


Clicking this close button will close the following views:
"Synoptic Hierarchy"
"System Configuration Browser"
Do you still want to close all the views in this group?
Yes
No

The editor will warn you of the closing of multiple views.

This concludes the lesson in the USS Editors docking mechanism.

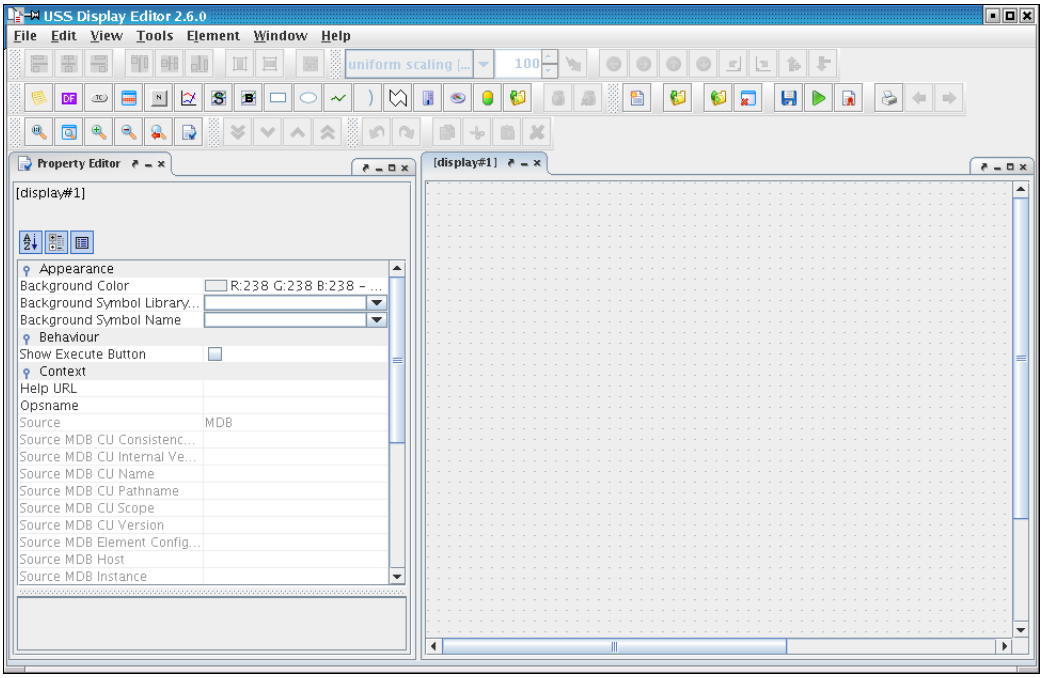
3.3.3 Tool views of the editor

The USS Editor features a multitude of tool-views, default views are opened in their latest position and size. The editor menu gives the easiest access to the tool-views.

1. Property Editor, show current selected item's properties and allow to edit them if they are editable
2. Consistency Checker, show the results of latest consistency check for current display, information, warnings and errors are displayed
3. Synoptic Hierarchy, show all open displays, as well as the content of the USS project-folder, and can be used for easy navigation, when multiple displays are open
4. Symbol Library, show the currently open library and allows to change library and select symbols
5. System Configuration Browser, show the content from the currently SCOE file, and can be used for easily adding End-items to displays
6. View Settings, show display settings like grid configuration etc. for the currently selected display view

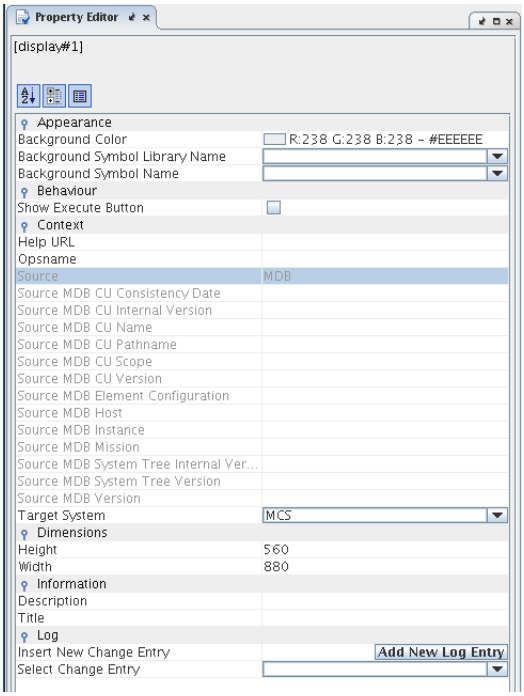
3.3.3.1 Lesson in tool views

1. Start by opening the editor via installed icon, when open close all tool-views and open Property Editor from menu: Views | Property Editor.



Editor started with default empty display created, property editor showing (layout might differ).

2. Undock the Property Editor.



The Property Editor is undocked and it is showing the open display settings. Showed for a display is the following categories:

(a) Appearance - Basic appearance of display like background colours

(b) Behaviour - Only behaviour setting for display

(c) Context - Information about displays context and references

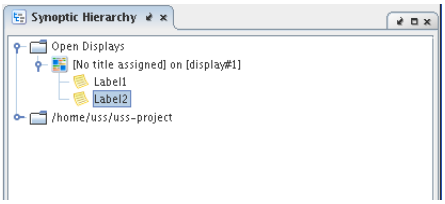
(d) Dimensions - Dimensions of display: width and height

(e) Information - Description and title information

(f) Log - Logging information for simple revision control

3. Close the Property Editor by clicking the close icon in the undocked view.

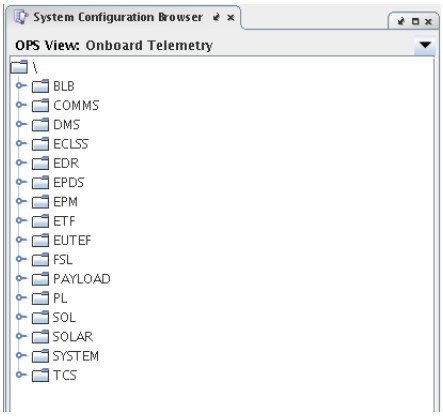
4. Open the Synoptic Hierarchy from menu: Views | Synoptic Hierarchy and undock it.



The Synoptic Hierarchy is undocked and it is showing the open display selected, as well as the displays (not-opened) saved in the USS project-folder. The Synoptic Hierarchy can be used to select displays and elements within, as well as it easily gives access to the project displays.

5. Close the Synoptic Hierarchy by clicking the close icon in the undocked view.

6. Open the System Configuration Browser from menu: Views | System Configuration Browser and undock it.



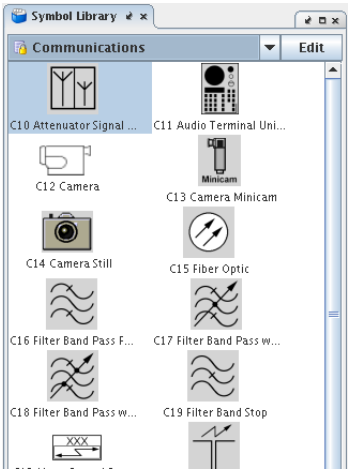
The System Configuration Browser is undocked and it is showing the content of the SCOE file, different OPS and Path views exist as well as categories:

(a) Onboard Telemetry
(b) Ground Telemetry
(c) Onboard Commands
(d) Ground Commands
(e) Onboard Events
(f) Ground Events

The System Configuration Browser can be used to SCOE content to the displays without hard configuration task.

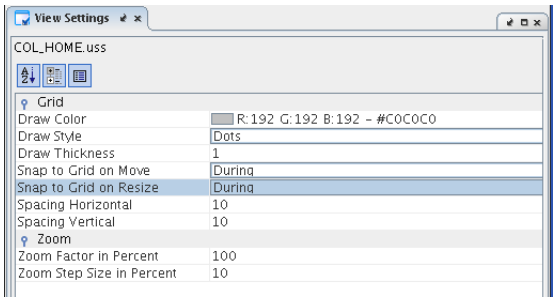
7. Close the System Configuration Browser by clicking the close icon in the undocked view.

8. Open the Symbol Library from menu: Views | Symbol Library and undock it.



The Symbol Library is undocked and it is showing the open symbol library selected. The Symbol Library can be used to select libraries and symbols as well as it provides access to adding and editing symbol libraries and their symbols.

9. Close the Symbol Library by clicking the close icon in the undocked view.
10. Open the View Settings from menu: Views | View Settings and undock it.



The View Settings is undocked and it is showing the open view settings of the display selected. The View Settings can be used to alter the grid and appearance of the display in the editor. It has the same layout and features as the Property Editor Showed for a display is the following categories:

(a) Grid - Basic appearance of Grid in the display view, as well as behaviour off elements when they are being moved or resized

(b) Zoom - Zoom settings

11. Close the View Settings by clicking the close icon in the undocked view.

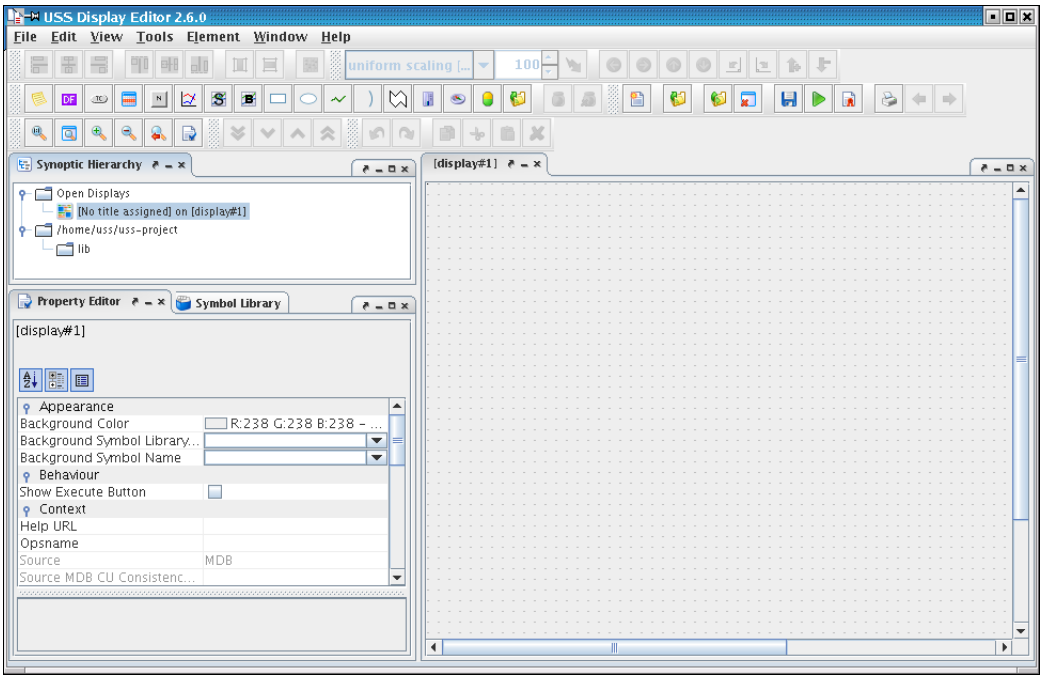
This concludes the introduction to the editors tool-views.

3.3.4 Editor Print Facility

The USS Editor features a standard printing feature, which gives access to basic layout and scaling of the printout.

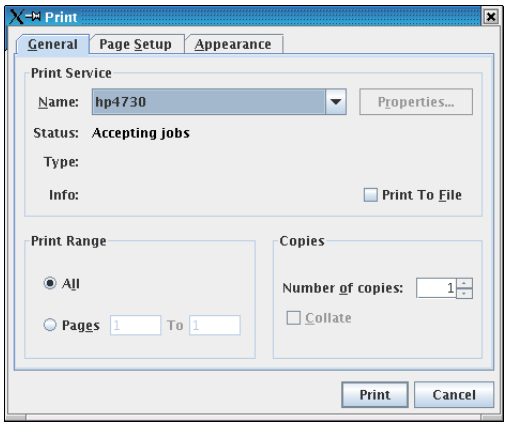
3.3.4.1 Lesson in printing of displays

1. Start by opening the editor via installed icon.



Editor started with default empty display created.

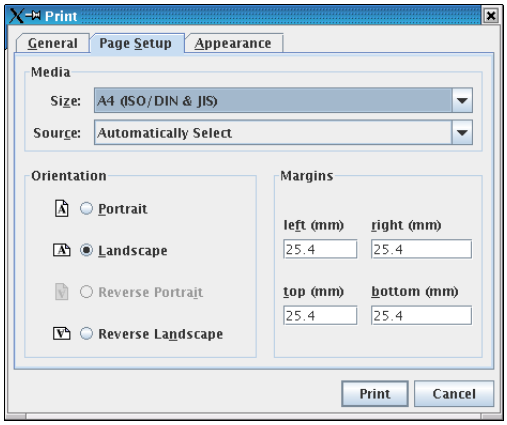
2. In the editor select from menu: File | Print



The editors print dialog opens with default printer selected. The General tab the following print properties can be selected:

(a) Name - Selects Printer
(b) Properties - Properties for printer if available
(c) Print To File - Check if print should go to a post-script file
(d) Print Range - Select the range of pages to print
(e) Copies - Number of copies and how to handle multiple copies

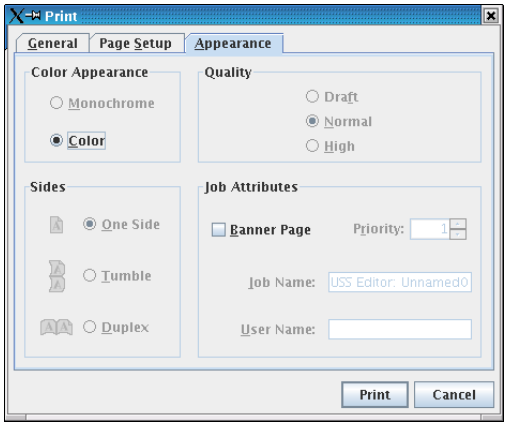
3. Click the tab: Page Setup to show further properties



The Print dialog shows the Page Setup properties for printing, which are described below:

(a) Size - Selects Paper size in printer
(b) Source - Select paper source, i.e. tray in printer
(c) Orientation - How the print is oriented on the paper
(d) Margins - Margins on the paper

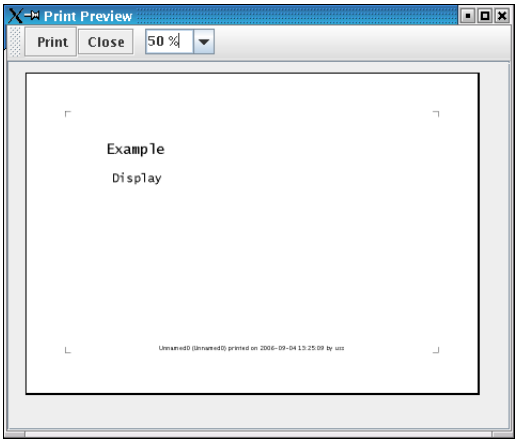
4. Click the tab: Appearance to show further properties and check properties: Banner Page



The Print dialog shows the Appearance properties for printing, which are described below, if available:

(a) Color Appearance - Sets if print should be in color
(b) Quality - Select output quality
(c) Sides - Select the pages arrangement
(d) Job Attributes - Selects job attributes, like banner page and priorities

5. Click print to print display.
   The Print is printed on the configured printer with a banner page first.
6. In the editor select from menu: File | Print Preview



The Print Preview dialog shows the expected print layout.

This concludes the lesson in the USS Editors printing mechanism.

3.4 Quick Tutorial

This tutorial section explains the preparations needed in order to run both USS executor and USS editor. After this it gives a short tutorial on USS editor, which shows how to create a new ground ops (Satmon like) display, the converting of GWDU ground displays in batch operation and how to check GWDU displays for consistency. It finishes with a tutorial on USS executor, which explains how to connect and disconnect a display, shows direct commanding of FWDU displays, shows a GWDU display as well as imported Satmon displays samples and PCS display samples and shows the commanding via MCS Tools.

3.4.1 Preparation

This section explains the preparations needed in order to run both USS executor and USS editor.

1. Start MCS Console
   (a) Start Task Selector by typing **ts&** into the command shell.
   (b) Start TSCV and press continue on all warnings that may appear.
   (c) Start MCC_Client.
   - Shutdown MCC_Client if already running.
   - Start MCC_Client and wait until it reaches active status.

   (d) Start HCI Online Test Control.
   - In the HLCL shell type **init_client** to start TM acquistion.
2. Keep "DOWNLINK_STATUS" GWDU display open (for later USS side-by-side test).
3. Switch to an empty desktop workspace.
4. Open a task selector by opening a shell and typing **ts&** into it.

3.4.2 Editor Tutorial

This section gives a short editor tutorial. It shows how to create a new ground ops (Satmon like) display, the converting of GWDU ground displays in a batch operation and how to check GWDU displays for consistency.

3.4.2.1 Creating new ground ops (Satmon like) display

3.4.2.1.1 Setting project root folder and MDB configuration

- Close all open displays, this is a requirement to change the project root folder
- Choose **Edit > Preferences**
- In the popup dialog choose **Project** tab in the tree structure on the left.
- Click **Browse** button to set the **Root folder** to /home/<user>/uss-project .

3.4.2.1.2 Opening and saving displays from and to MDB

- Start my making a new display from menu choose: File | New
- Add a label by choosing from menu: Element | Add | Label, and add a text by editing label text in Property Editor (open it if not open by choosing from menu: Views | Property Editor)
- Save display by choosing from menu: File | Save. In dialog save display to location and display-name (capitalize name of display): [USS-Project-folder]/[CU-DIRECTORY]/APM/COMMON_TEST_SUPP/G
- Press OK to save dialog, and choose from menu: File | MDB | Add to MDB
- Wait for operation to finish and press OK to confirmation dialog. Displays are now saved in the MDB.
- Close all displays by choosing from menu: File | Close all, displays might need to be save or cancelled.
- Choose **File > MDB > MDB Browser (sync/lock-for-edit).**
- Select \APM\COMMON_TEST_SUPP\GRD_DATA\USS\EDITOR\MANUAL_TESTS\TUTORIAL in the browser.
- Right-click then select **Sync** from the Version Control popup-menu and wait for synchronization to complete.
- Click **Apply**.
- In the **synoptic hierarchy** on the right side select root folder (the one which you have configured in Setting project root folder and MDB configuration.
- Navigate to \APM\COMMON_TEST_SUPP\GRD_DATA\USS\EDITOR\MANUAL_TESTS.
- Right-click on TUTORIAL.uss and select **Open Display**. TUTORIAL opens in editor view.
- Right-click on display. Select **Lock-for-Edit on MDB**.
- Edit display.
- Choose **File > Save** display.
- Right-click on display. Select **Save in MDB**.

3.4.2.1.3 Editing displays

- Creating Elements i.e. labels, fields etc.
  1. Choose **Elements > Add** .
  2. Select Label or Data Field.
- Changing properties of elements.
  1. Right-click on newly created element (label, data field etc.)
  2. From the popup dialog select action you would like to perform (e.g. align, move, order, rotate or scaling).
- Drag and drop an item from the system configuration browser onto the display.
- Drag and drop multiple items from the system configuration browser onto the display.

3.4.2.1.4 Previewing a display

- Save current display by choosing **File > Save**.
- Choose **Tools > Preview Display...**.
- Watch display open in a previewer, the slider (in the bottom of window) can be used to see any dynamic contents.
- Close previewer window.

3.4.2.2 Converting GWDU ground displays as batch operation

- Choose **Tools > Run Batch Operation...** .
- Click Browse and set **source directory** to <USS base directory>/examples/import/gwdu.
- Click Browse and set **destination directory** to /home/<user>/uss-project .
- Click **Convert**: The batch process starts.
- Click Close.
- After conversion: Close and reopen synoptic hierarchy.
- Open a converted USS display in /home/<user>/uss-project/mcs/... for example (/APM/COL_CC/MCS_OPS/CON

3.4.2.3 Checking GWDU display for consistency

- Open /APM/COL_CC/MCS_OPS/CONFIG/SYNOPTICS/MCS/DOWNLINK_STAT.uss.
- Choose **Tools > Consistency Check...**.
- In the **Consistency checker** report displayed at the bottom, double click line with error item, item gets selected in display.

3.4.3 Executor Tutorial

This section gives a short tutorial on how to use the executor. It shows how to connect to and disconnect displays from MCS. It explains direct commanding from a FWDU display as well as commanding via MCS Tools. It explains how to show a GWDU display, imported Satmon display and PCS display samples.

3.4.3.1 Connecting and opening displays

- Choose **Options > MCS Connection...** to connect to MCS.
- In the popup dialog enter the correct connection data for service (e.g. CIS) and host (e.g. localhost) and port (e.g. 7060).
- Then click Connect.
- Open **Col Home** display.

3.4.3.2 Show direct commanding (from FWDU display)

- Open /APM/FLTSYS/OPS/SYNOPTICS/ECLSS/CFA1 display.
- Choose **Options > Preferences...**. Enable direct commanding.
- In the popup window select Commanding in the tree structure on the right side.
- Check Enable Commanding.
- Check Direct Commanding.
- Click OK.
- Right-click in white box under Fan Speed.
- In the popup dialog select Line Graph.
- Leave line graph window open.
- In the display click on Pwr Off command button.
- Click **Execute** button in the bottom of the display: Watch command feedback in response panel. Pwr changes to Off, Fan Speed drops to 0 rpm. Watch line graph changing value.

3.4.3.3 Show GWDU display

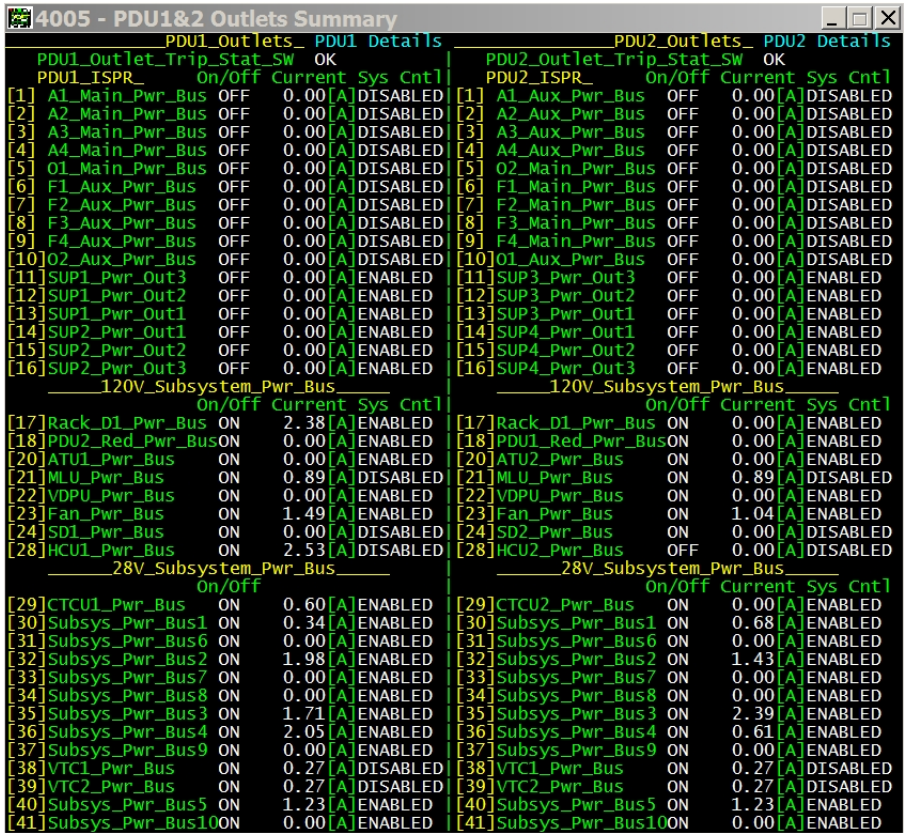
- Open APM/COL_CC/MCS_OPS/CONFIG/SYNOPTICS/MCS/DOWNLINK_STAT.gwdu to show updated telemetry
- Click **TM Packet Status...** button
- Show monitoring
- Show display navigation
- Show parameter properties
  1. Right-click on parameter.
  2. In the popup dialog select Properties.
  3. Click Close.
- Show display search for selected parameter
- Right-click on parameter.
- In the popup dialog select Find Parameter In Other displays.
- Click Close.

3.4.3.4 Show commanding via MCS Tools

- Choose **Options > Preferences**.
- Select Commanding in the tree structure on the right.
- Check Enable Commanding.
- Uncheck Direct Commanding.
- In Command Target select or enter **Manual Stack@mcs-cc-3**.
- Click OK.
- Start MCS Tools: **Task Selector > Onboard Commanding**.
- In MCS Tools select **Options > Preferences**. Set remote commanding to **Manual Stack@mcs-cc-3**.
- Select **CFA1 Deactivation** and **Execute**.
- Select row in manual stack, **Enable, Activate**.
- Select **CFA1 Activation** and **Execute**.
- Select row in manual stack, **Enable, Activate**.

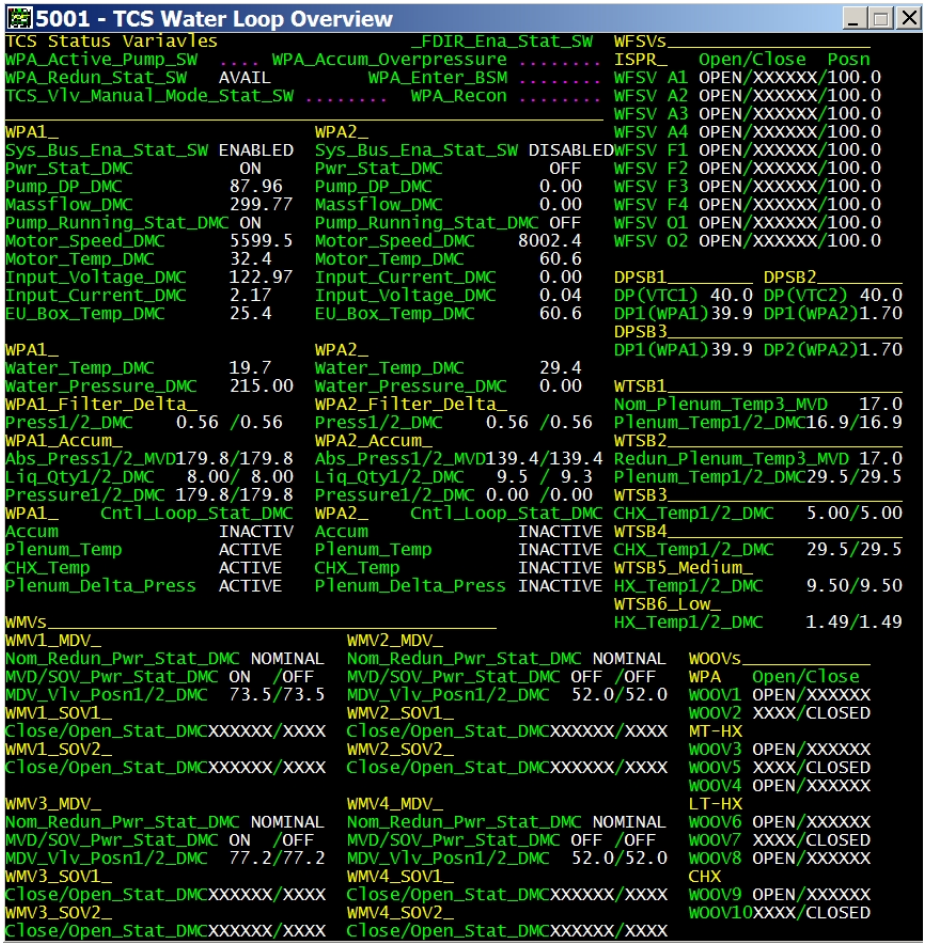
3.4.3.5 Show imported Satmon display samples

- Choose **File > Open...**.
- Open $USS_HOME/examples/import/uss/satmon/4005.uss.
- Compare with screenshot of original display:



- Open $USS_HOME/examples/uss/satmon/5001.uss

- Compare with screenshot of original display:



3.4.3.6 Show PCS display samples

- Choose **File > Open Display.**
- Open $USS_HOME/share/displays/pcs/xml/Columbus_ECLSS.xml
- Open $USS_HOME/share/displays/pcs/xml/Col_Air_Sensors.xml
- Open line graph on **Liquid Carryover Sensor 1**.

3.5 Making a Display

This tutorial section explains the preparations needed in order to run both USS executor and USS editor. After this it gives a short tutorial on USS editor, which shows how to create a new ground ops (Satmon like) display, the converting of GWDU ground displays in batch operation and how to check GWDU displays for consistency. It finishes with a tutorial on USS executor, which explains how to connect and disconnect a display; shows direct commanding of FWDU displays, shows a GWDU display as well as imported Satmon displays samples and PCS display samples and shows the commanding via MCS Tools.

3.5.1 Introduction

This lesson will go through the a basic display in the USS Editor. You shall work with the following subjects:

1. Making of a display
2. Adding elements

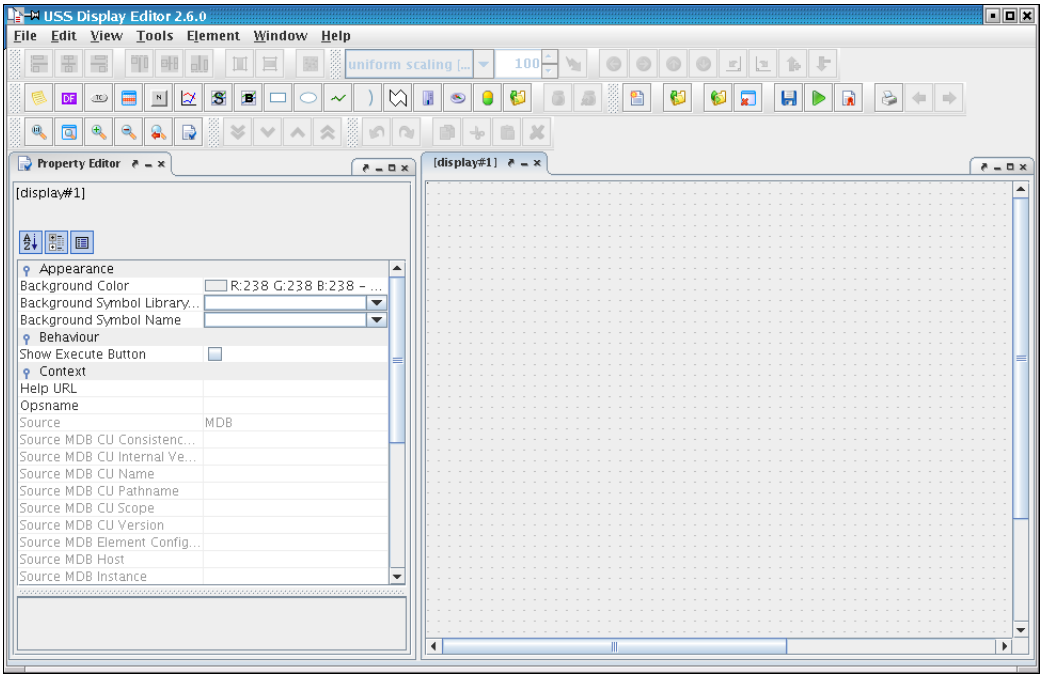
3. Previewing a display

3.5.2 Making display

The USS Editor can make display in the USS XML format via its GUI.

3.5.2.1 Lesson in making a display

1. Start by opening the editor via installed icon.



Editor started with default empty display created, three tool views open: Synoptic Hierarchy, Property Editor and Symbol Library. This is an example of start-up layout, the editor saved basic layout and which tools are open from previous editor closing.

2. Add a new display by selecting from menu: File | New

   A new display is opened.

3. Add a label by choosing from menu: Element | Add | Label. Click on label and edit text of label and resize the label by pulling the corner of the label.

   The New display has a label with your added text.

4. If Property Editor is not open, open it by choosing: Views | Property Editor

   Property Editor Open

5. In the Property Editor check the Label Text Auto size.

   The Label changes text size to match the size of the label.

6. Click on the in the display area outside the label area

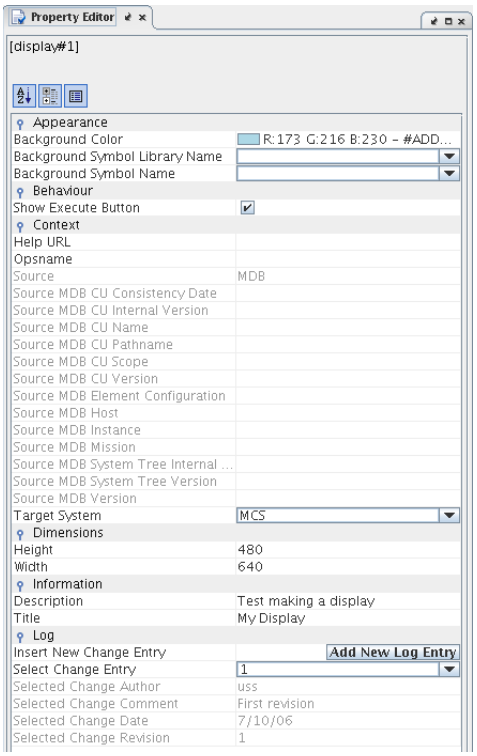
   Display is selected, and Property Editor shows the properties of the display.

7. Undock the Property Editor and edit the following properties by clicking in the field for the properties int Property Editor and editing:

   (a) Background color: Click button: '...' and add from color dialog a light blueish color.

   (b) Check the property: Show Execute Button

   (c) Fill in the properties in the category: Information

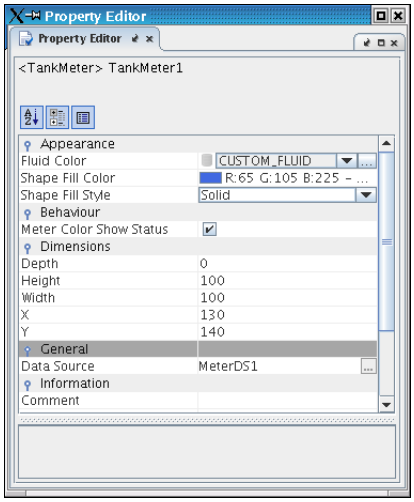


   The Property Editor shows the edited properties for the display.

8. Add a Tank meter, a Telecommand button and a Rectangle from the editor menu: Element | Add | ...

   Display contains a Label, Tank meter, Telecommand button and a Rectangle.

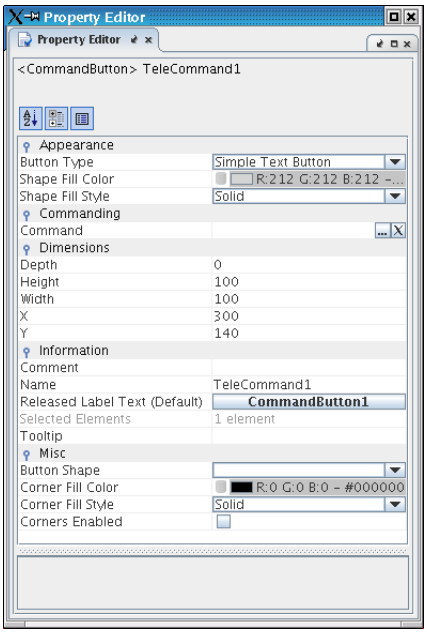
9. Click the added Tank meter to select it, edit it properties in the Property Editor to match:



Result after editingTank Meter changes with it properties

> **TIP**
>
> In the Property Editor, the property: Data Source contains a button: '...', which gives access to the data source editing dialog

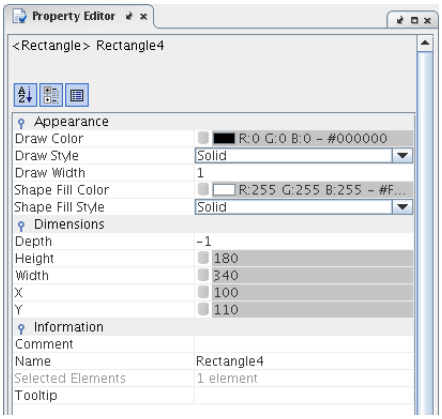
10. Click the added Telecommand to select it, edit it properties in the Property Editor to match:



Result after editingTelecommand has changed.

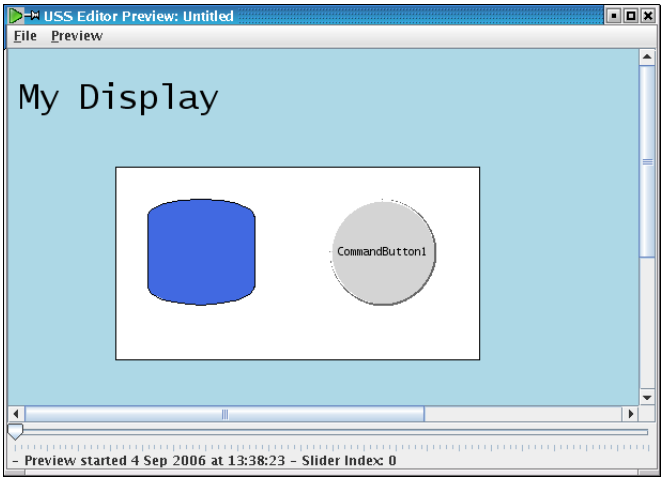
> **TIP**
>
> In the Property Editor, the property: Command contains a button: '...', which gives access to the command editing dialog

11. Click the added Rectangle to select it, edit it properties in the Property Editor to match:

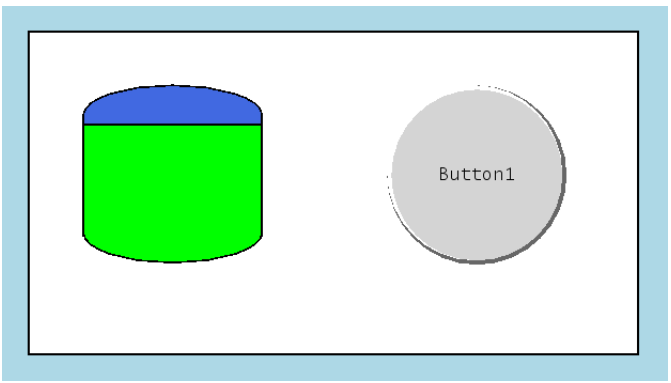


Display is edited.

12. Now you should preview you created display to get a better feel for it appearance and test its behaviour. Select from menu: Tools | Previewer ... Display is opened in the previewer frame:

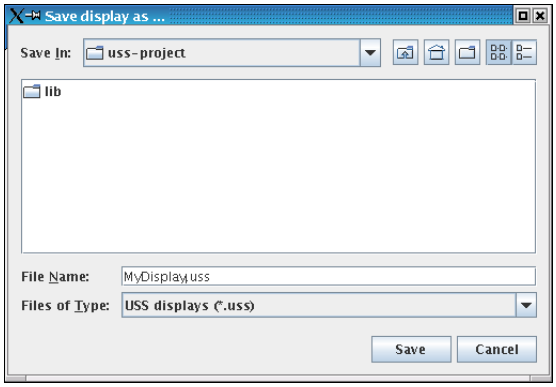

USS Editor Preview: Untitled
File Preview
My Display
CommandButton1
- Preview started 4 Sep 2006 at 13:38:23 - Slider Index: 0

13. Now you should use the slider at the bottom of the Previewer and see the Tank meter level move Previewer frame snap-shoot:

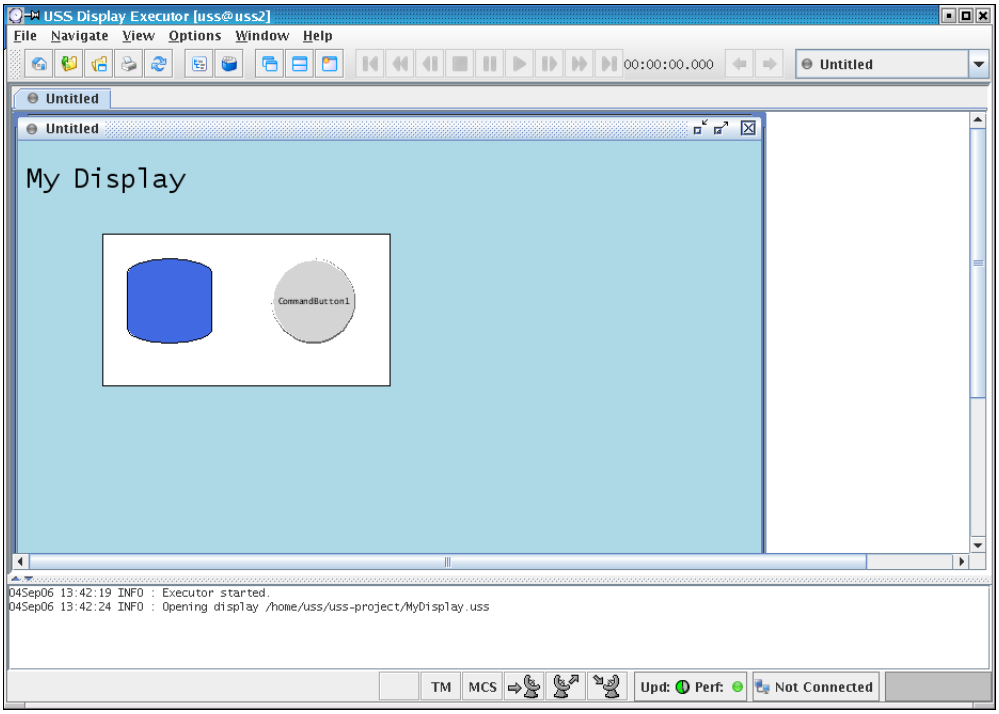


14. Close the Previewer frame.

15. Save display by selecting from menu: File | Save Save dialog is opened, where you can enter display file-name



16. Now try opening the display in the USS Executor select from menu: Tools | Executor Display is opened in the USS Executor, which is the real execution application for the display:



This concludes the lesson in the USS Editors docking mechanism.

3.6 Import existing displays

The USS Editor can import displays from the following formats:

1. FWDU displays
2. GWDU displays
3. NASA PCS displays
4. Satmon displays

Imported display definitions become USS displays meaning that they will have the USS XML-based file format and the .uss file extension. The imported displays cannot be exported back into the legacy display formats.

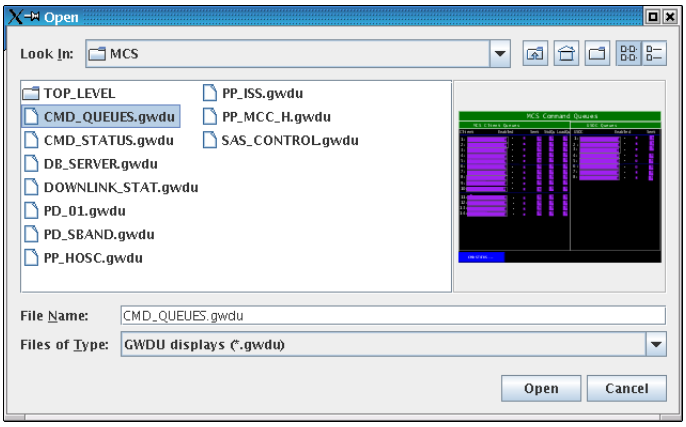
3.6.1 Lesson in importing a display

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout

2. Open / import a GWDU display by selecting from menu: File | Open

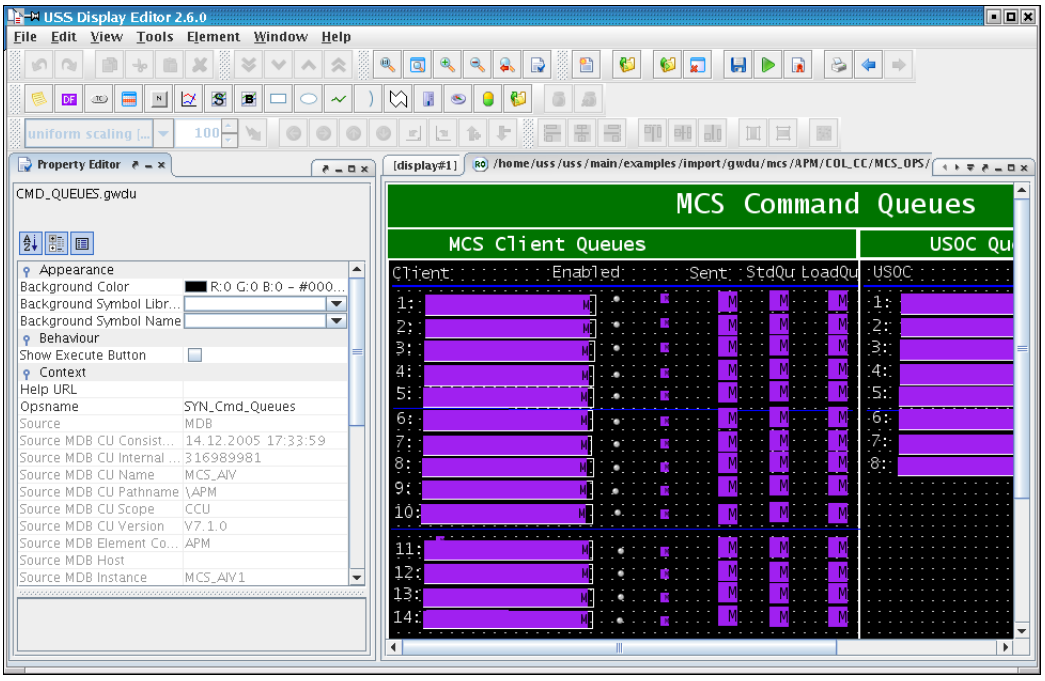
   The file open dialog is now open.

3. Browse via the open dialog to the display: CMD_QUEUES.gwdu Remember to change File of Type to All Files CMD_QUEUES.gwdu can be found under path: <USS_INSTALL_DIR>/examples/import/gwdu/r



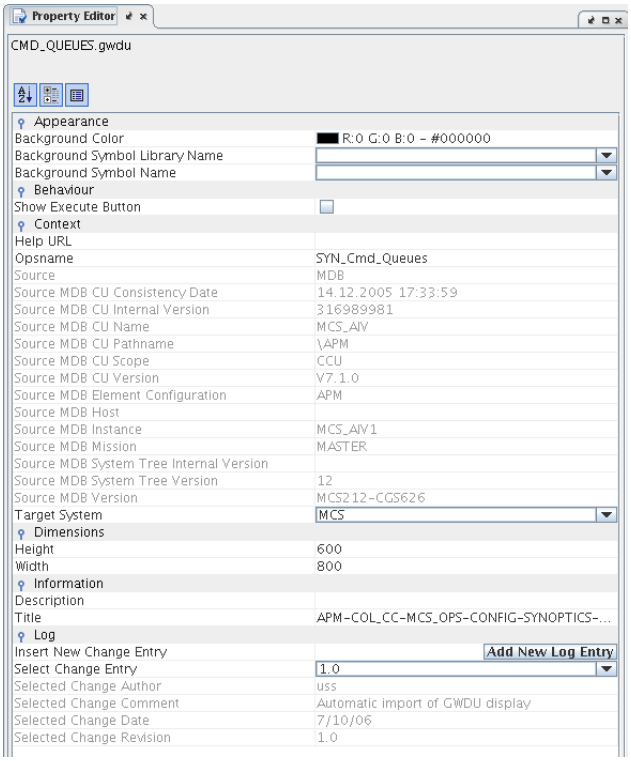
   The import display selected, preview shown.

4. Click open to open display.



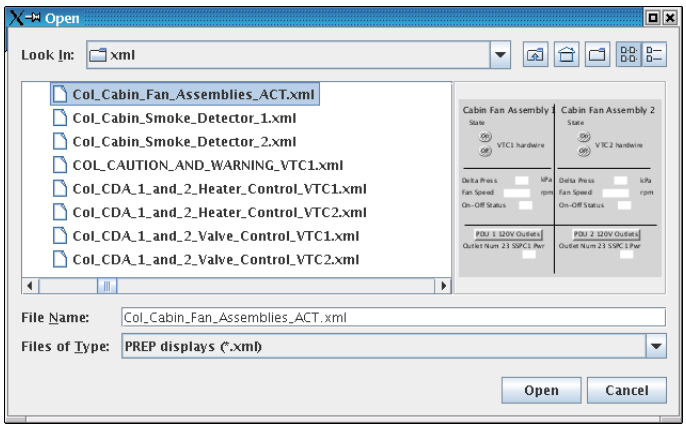
   The imported display is shown shown in the editor, the import process is seamless and a conversion is done between GWDU and USS format. The display can now be edited, and saved to the USS display format.

5. In the Property Editor check the display import information given by the Source information

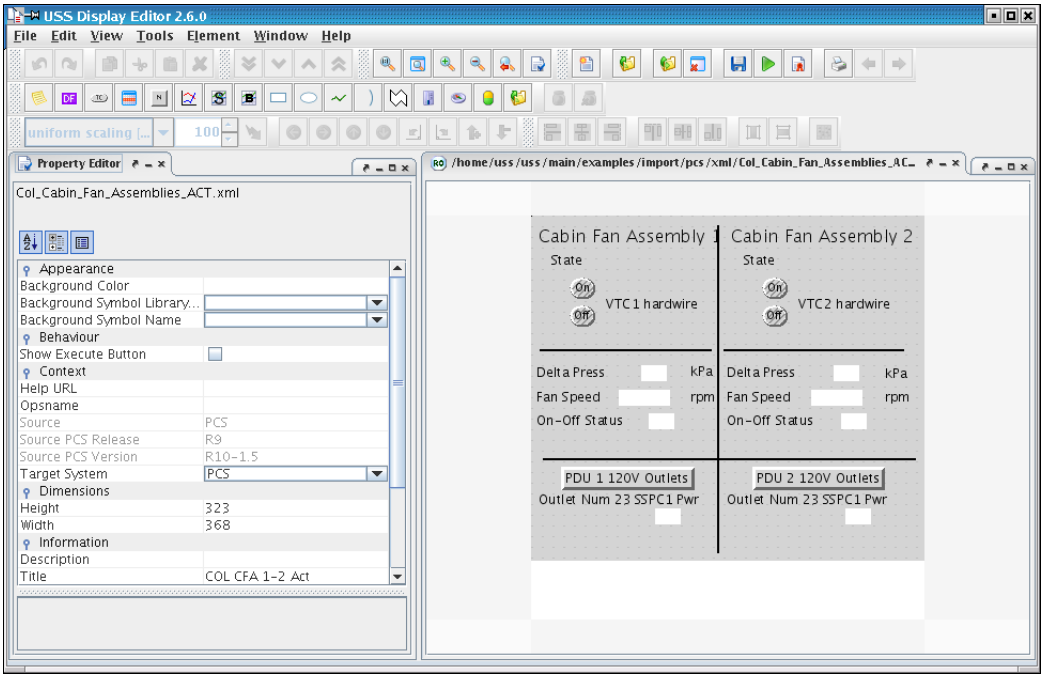


6. Close the imported GWDU display.

7. Now import a PCS display with path: <USS_INSTALL_DIR>/examples/import/pcs/xml/Col_Cabin_FanAssemblies

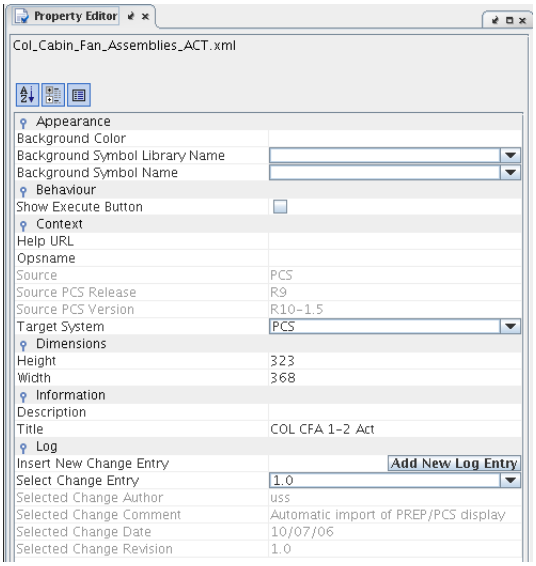


8. Click open to open display.



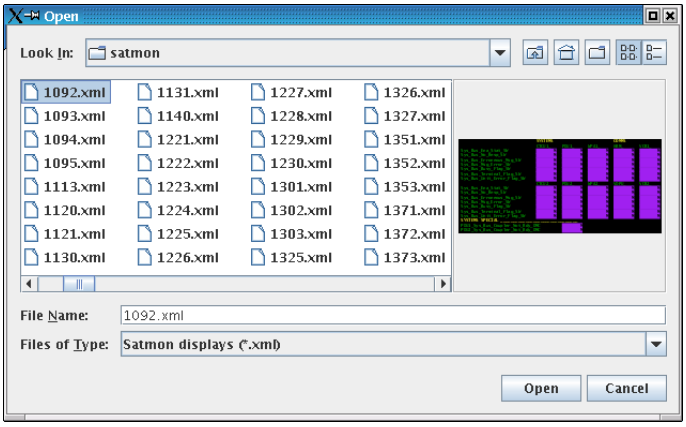
The imported display is shown shown in the editor. The display can now be edited and saved to the USS display format.

9. In the Property Editor check the import information.

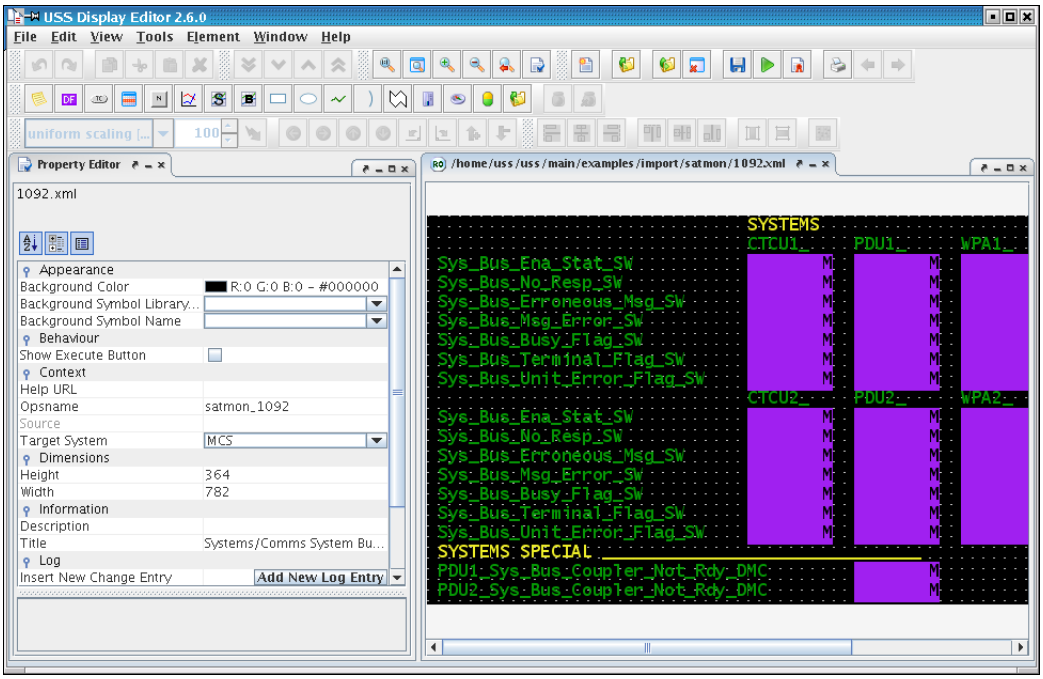


The Property Editor shows the properties for the imported display.

10. Now import a Satmon display with path: <USS_INSTALL_DIR>/examples/import/satmon/1092.xml

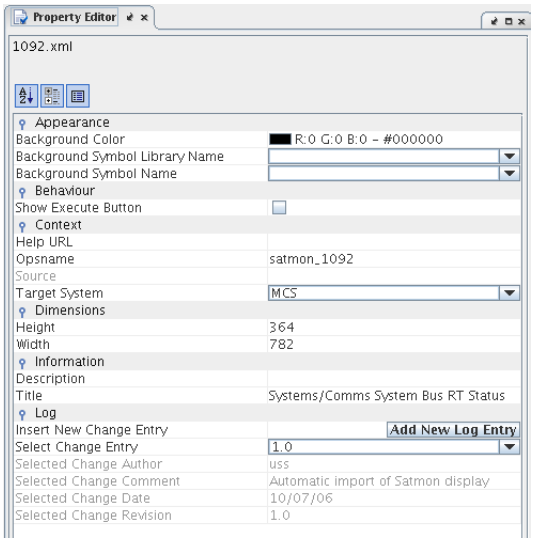


11. Click open to open display.



The imported display is shown shown in the editor. The display can now be edited and saved to the USS display format.

12. In the Property Editor check the import information.



The Property Editor shows the properties for the imported display.

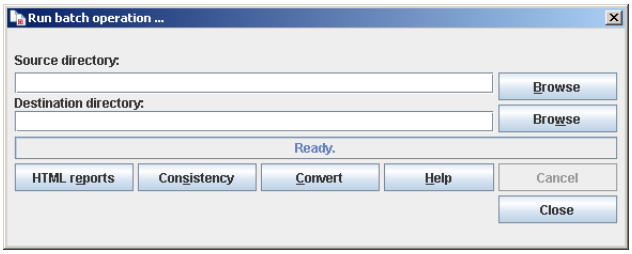
This concludes the lesson in the USS Editors import mechanism. The USS Editor and Executor use the same import mechanism.

3.6.2 Lesson in batch importing displays

The USS Batch operation tool/dialog can be used to performe a operation on multiple displays.
Batch operations:

- HTML reports - A HTML Report contains a picture of display and display information
- Consistency - A comparison of display end-item references with actual SCOE file, set in Editor preferences
- Convertion - A convertion of a non-USS display format to the USS display format

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout
2. Open batch dialog by selecting from menu: Tools | Run batch operation ...



The batch dialog is now open

3. Browse via source directory browse button to USS installation directory: <USS_INSTALL_DIR>/examples/imp
4. Browse via destination directory browse button to temporary files directpry: /tmp
5. In the Run batch operation dialog click button: Convert, to start the convertion of PCS-displays.

The batch operation starts and progress is shown in bar

6. The batch operation will continue until all in source directory displays are converted.

The batch operation completion is shown in bar and re-enabling of buttons

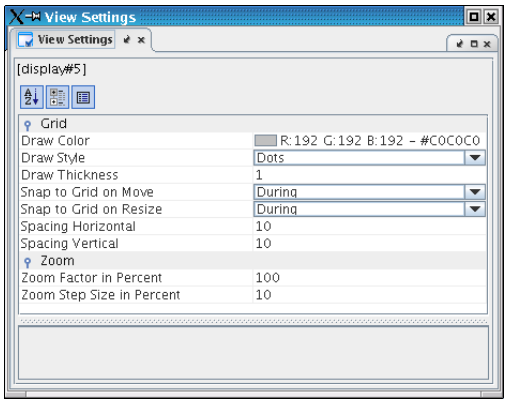
This concludes the lesson in the USS Editors batch import mechanism. The USS Editor and Executor use the same import mechanism, but batch operation are mainly awailable through the Editors batch tools.

3.7 GUI Elements

This lesson takes you through the major GUI elements of the USS Editor namely the View Settings and the Preferences.

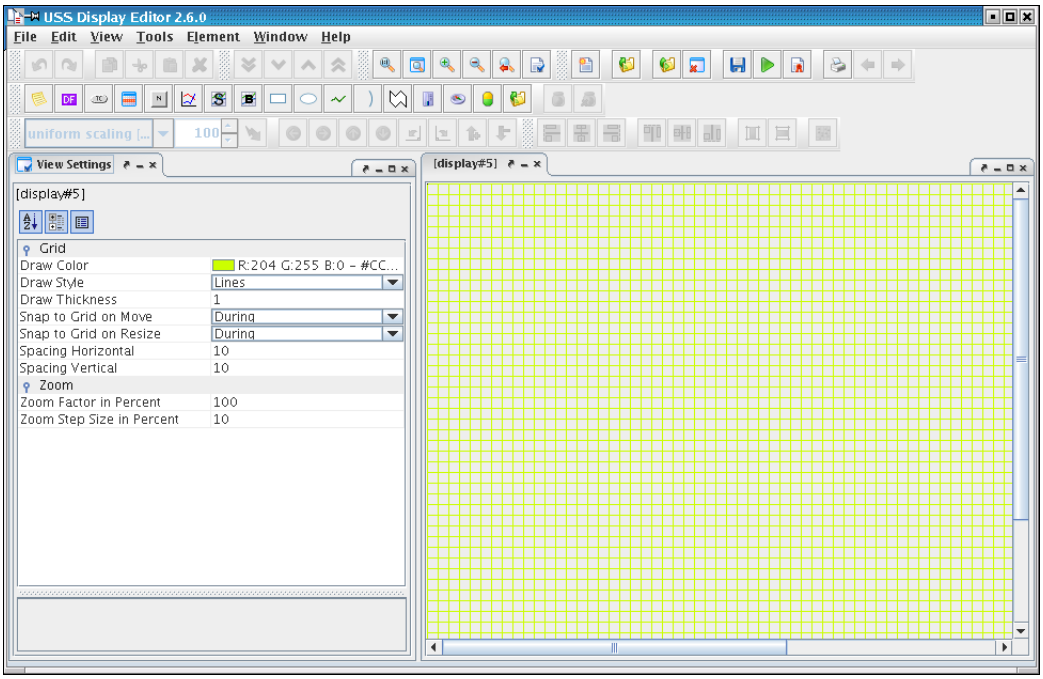
3.7.1 Lesson in using the GUI elements (View Settings and Preferences)

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout
2. Close all tool views and open the View Settings



A Display is open with the View Settings showing. The View Settings are not saved with the displays, the properties are non-persistent and only for viewing in the Editor. They provide a help when editing.

3. Change the grid colour to draw style to yellow and the draw style to Lines.



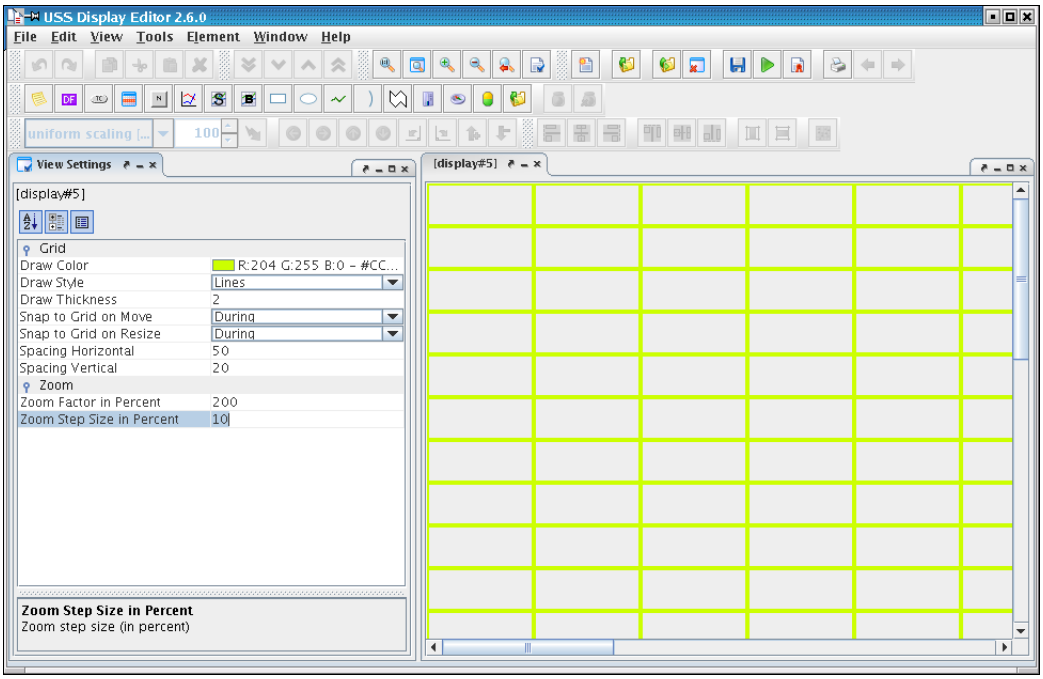
The Display grid colour change as well as the draw style, the grid is now fully drawn.

4. Now change the following properties:

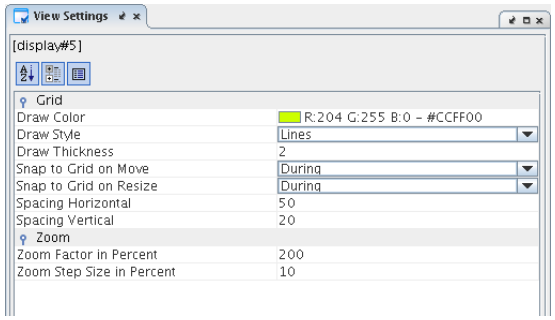
   (a) grid spacing to 50/20 (Horizontal/Vertical)

   (b) set the Zoom factor to 200 %

   (c) grid draw thickness to 2 pixels

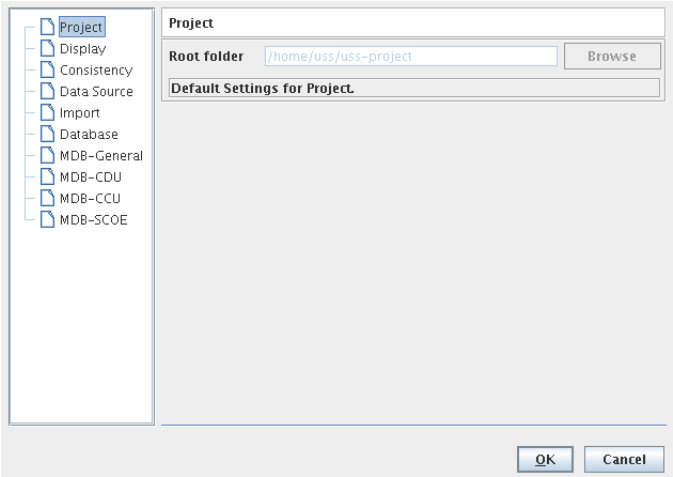

USS Display Editor 2.6.0
File Edit View Tools Element Window Help
View Settings
[display#5]
Grid
Draw Color R:204 G:255 B:0 - #CC...
Draw Style Lines
Draw Thickness 2
Snap to Grid on Move During
Snap to Grid on Resize During
Spacing Horizontal 50
Spacing Vertical 20
Zoom
Zoom Factor in Percent 200
Zoom Step Size in Percent 10
Zoom Step Size in Percent
Zoom step size (in percent)

5. Undock the View Settings and to get a better view of the settings changed.



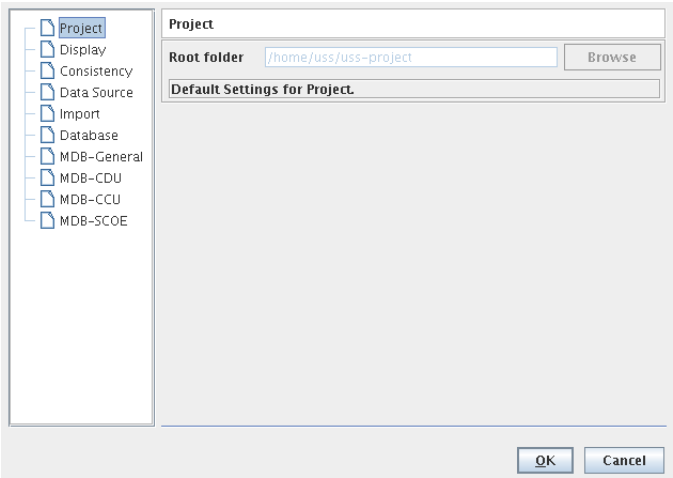
Walk-through of the Editor Preferences

1. Open the editor preferences by selecting from menu: Edit | Preferences.



The Editor preferences opens, it is arranged after categories (to the left) and each category contains a group of properties. No properties change in the system before the OK-button is pressed, so you can change properties without effect, as long as you do not press OK.

2. Click on the Project category

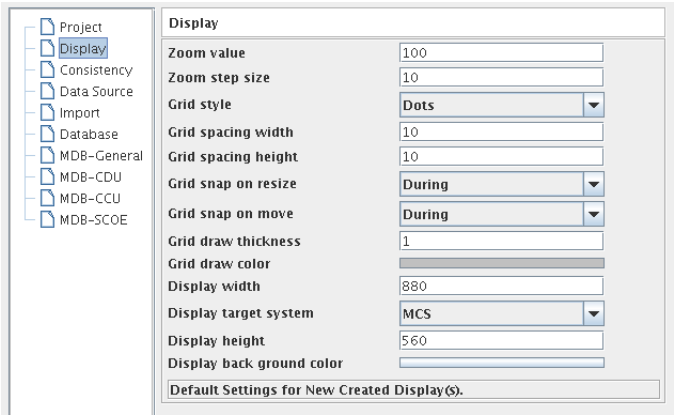


The project category contains the following properties:

   (a) Root Folder

   The Root Folder defines where the uss-project is located, that is were MDB files are synchronized to, and provide common location for displays.

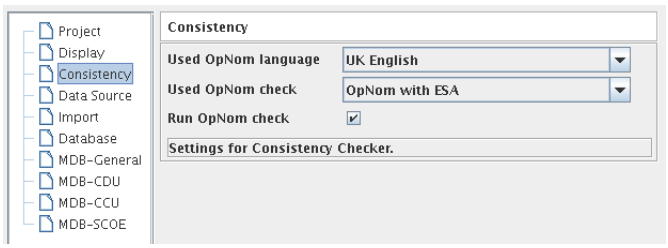
3. Click on the Display category



The display category contains the following properties, which are used as default for new displays:

(a) Zoom value: Default zoom value in percent
(b) Zoom step size: Default step size in percent for zooming in and out.
(c) Grid Style: Default grid style choose between: Non, Dots, Dashed, Lines
(d) Grid spacing width: Default grid spacing width in pixels
(e) Grid spacing height: Default grid spacing height in pixels
(f) Grid snap on resize policy: Default behaviour of snapping when resizing an element in a display
(g) Grid snap on move policy: Default behaviour of snapping when moving an element in a display
(h) Grid draw thickness: Default grid thickness in pixels
(i) Grid draw colour: Default grid colour
(j) Display width: Default display width when creating anew display
(k) Display target system: Default display target system when creating anew display
(l) Display height: Default display height when creating anew display
(m) Display back ground colour: Default display back-ground colour when creating anew display

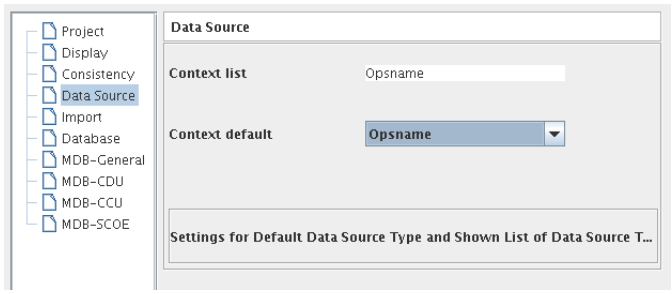
4. Click on the Consistency category



The consistency category contains the following properties:

(a) Used OpNom language: The language to use while spell checking in the consistency checker
(b) Used OpNom check: The OpNom standard to use while checking in the consistency checker
(c) Run OpNom check: Whether or not to run OpNom checking in the consistency checker

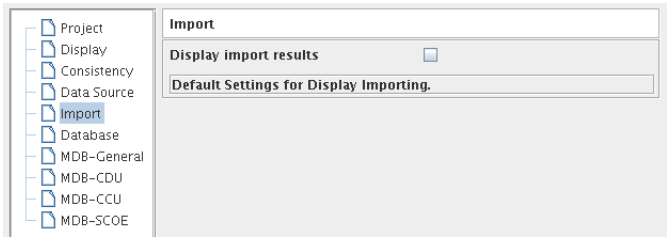
5. Click on the Data Source category



The Data Source category contains the following properties:

(a) Context List: The list of contexts to be available in the Editor
(b) Context default: The default context to use from the context list

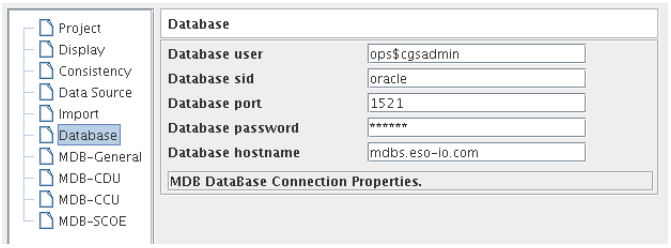
6. Click on the Import category



The Import category contains the following properties:

(a) Display import results: Whether or not the editor should show a summary of results when import/opening external format displays

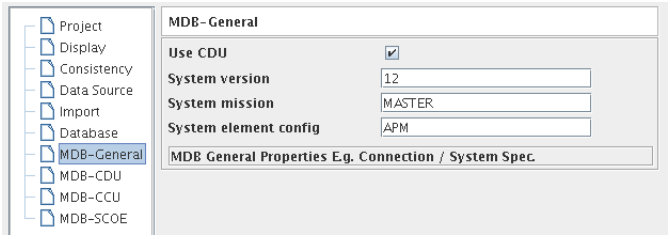
7. Click on the Database category



The Database category contains the following properties:

(a) Database user: The user to connect with
(b) Database SID: The system id (SID) of the MDB, normally oracle
(c) Database port: The TCP/IP port to connect to on the dB-server
(d) Database password: The dB password for the dB user specified
(e) Database host name: The host / db-Server to use

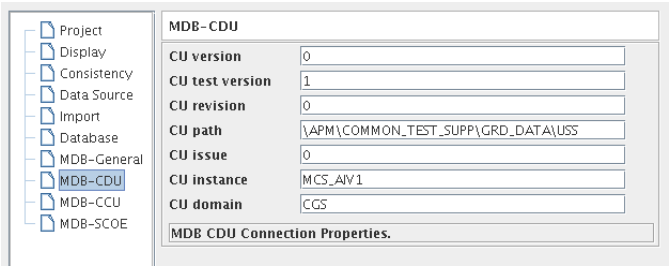
8. Click on the MDB-General category



The MDB-General category contains the following properties:

(a) Use CDU: Whether or not to use CDU (contra CCU)
(b) System Version: System version number
(c) System mission: System mission setting
(d) System element config: System element configuration

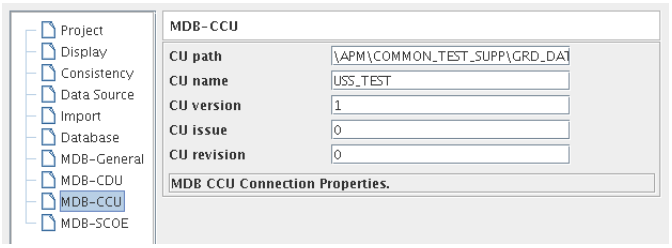
9. Click on the MDB-CDU category



The MDB-CDU category contains the following properties:

(a) Cu version: The CU version to use with CDU
(b) Cu test version: The CU test version to use with CDU
(c) Cu revision: The CU revision to use with CDU
(d) Cu path: The CU path to use with CDU
(e) Cu issue: The CU issue to use with CDU
(f) Cu instance: The CU instance to use with CDU
(g) Cu domain: The CU domain to use with CDU

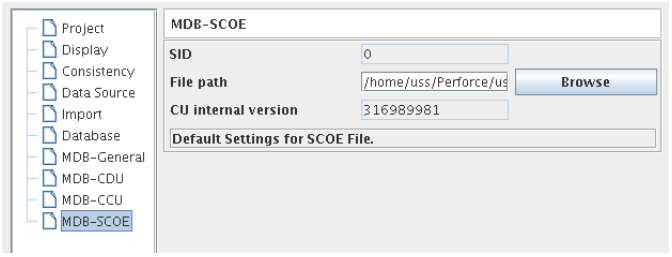
10. Click on the MDB-CCU category



The MDB-CCU category contains the following properties:

(a) Cu path: The CU path to use with CCU
(b) Cu name: The CU name to use with CCU
(c) Cu version: The CU version to use with CCU
(d) Cu issue: The CU issue to use with CCU
(e) Cu revision: The CU revision to use with CCU

11. Click on the MDB-SCOE category

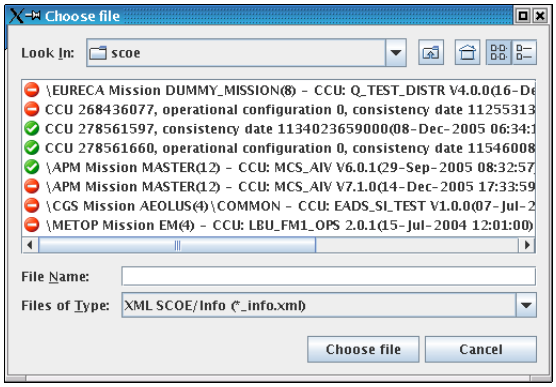


The MDB-SCOE category contains the following properties:

(a) SID: The System Id (SID) of the SCOE to use
(b) File path: The File path to the SCOE file

Shown in the category is also the SCOE internal CU version.

12. Click on the browse button to change SCOE file


Choose file
Look In:
scoe
\EURECA Mission DUMMY_MISSION(8) - CCU: Q_TEST_DISTR V4.0.0(16-De
CCU 268436077, operational configuration 0, consistency date 11255313
CCU 278561597, consistency date 1134023659000(08-Dec-2005 06:34:1
CCU 278561660, operational configuration 0, consistency date 11546008
\APM Mission MASTER(12) - CCU: MCS_AIV V6.0.1(29-Sep-2005 08:32:57
\APM Mission MASTER(12) - CCU: MCS_AIV V7.1.0(14-Dec-2005 17:33:59
\CGS Mission AEOLUS(4)\COMMON - CCU: EADS_SI_TEST V1.0.0(07-Jul-2
\METOP Mission EM(4) - CCU: LBU_FM1_OPS 2.0.1(15-Jul-2004 12:01:00)
File Name:
Files of Type:
XML SCOE/Info (*_info.xml)
Choose file
Cancel

Here the SCOE file can be selected, if you change it the System Configuration Browser will reload with the new information.

This concludes the lesson in the USS Editors preferences. Normally these settings are preset delivered from the system administrator.

3.8 Changing DQI Styles

The USS Editor and Executor use Data Quality Indicator definition to give information about the state of the data being processed.

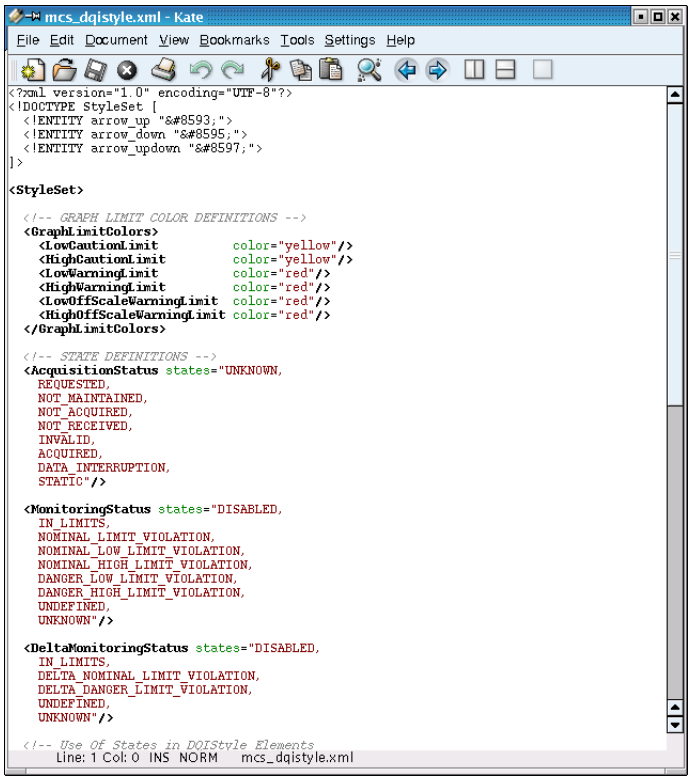
TIP

For thourough explanation of DQI in USS, see Data Quality Indication

3.8.1 Lesson in viewing and changing DQI files

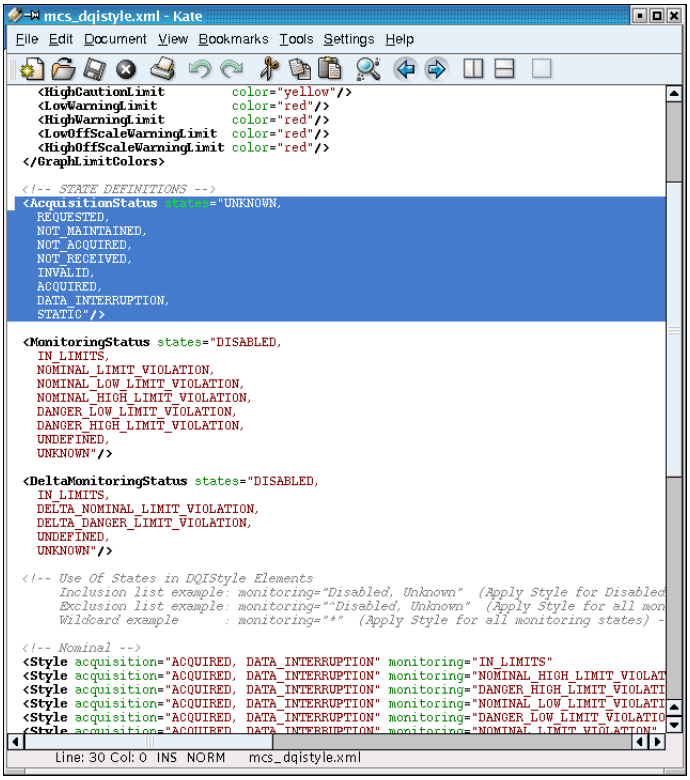
The lesson assumes the XML editor KXML-editor is installed.

1. Opening the KXML-editor and browser to the >USS_INSTALL</etc directory. Here open the file: mcs_dqistyle.xml



The KXML-Editor shows the content of the xml-formatted DQI file for the MCS target system. The XML tree hierarchy is shown on the left and the content of the selected node on the right, where the content can be edited.

2. Browse in the XML tree hierarchy to the node: AcquisitionStatus and select it to show the content.



Changing the values here will change the states used in the Editor Consistency Checker for the MCS target system.

3.8.2 Lesson in adding a DQI file

The lesson will show how to install a copy of the MCS DQI as a new DQI definition.

1. Opening terminal and enter the >USS_INSTALL</etc directory. Use command:

```
cd uss-2.6.0/etc
```

2. Copy the MCS DQI definition to a new file. Use command:

```
cp mcs_dqistyle.xml tst_dqistyle.xml
```

3. Check file is copied. Use command:

```
ls
```

4. Now open the file uss.properties also located in the etc directory.

```
kate uss.properties &
```

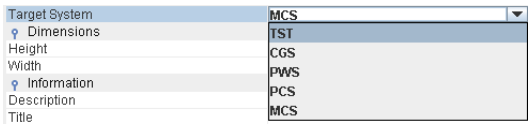
5. Make an entry for the new dqi style file by adding the following to lines in the DQI section of the properties file.

```
uss.view.dqistyle.target5 = TST
      uss.view.dqistyle.file5 = ${basedir}/etc/tst_dqistyle.xml
```

6. Save the properties file, now the USS application will, on next invoking, be aware of the new Display target and DQI style.

7. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout

8. In the empty new display created by the editor on default startup, try selecting the new DQI Style, i.e. display target in the proeprties of the display (properties in property editor tool view).



Editor shows the new DQI Style available as a display target in the property editor

This concludes the lesson in the USS DQI Style preferences. Normally these settings are preset delivered from the system administrator.

3.9 Create ASCII displays

The USS Editor can be used to create pure ASCII displays in the USS display format. ASCII display contains only character based elements.

3.9.1 Lesson in creating ASCII displays

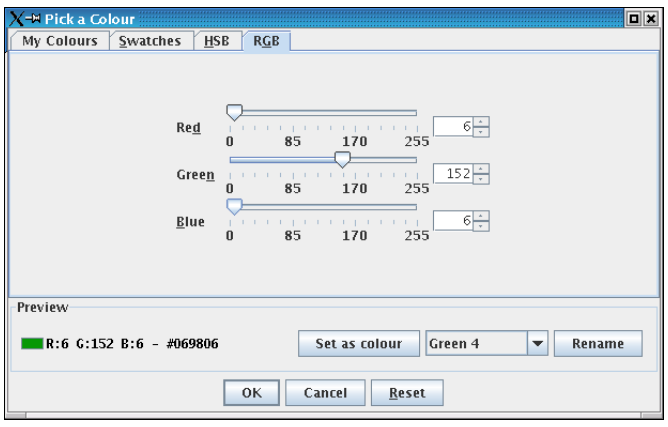
1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout

2. Add a label by selecting in the menu: Element | Add | Label

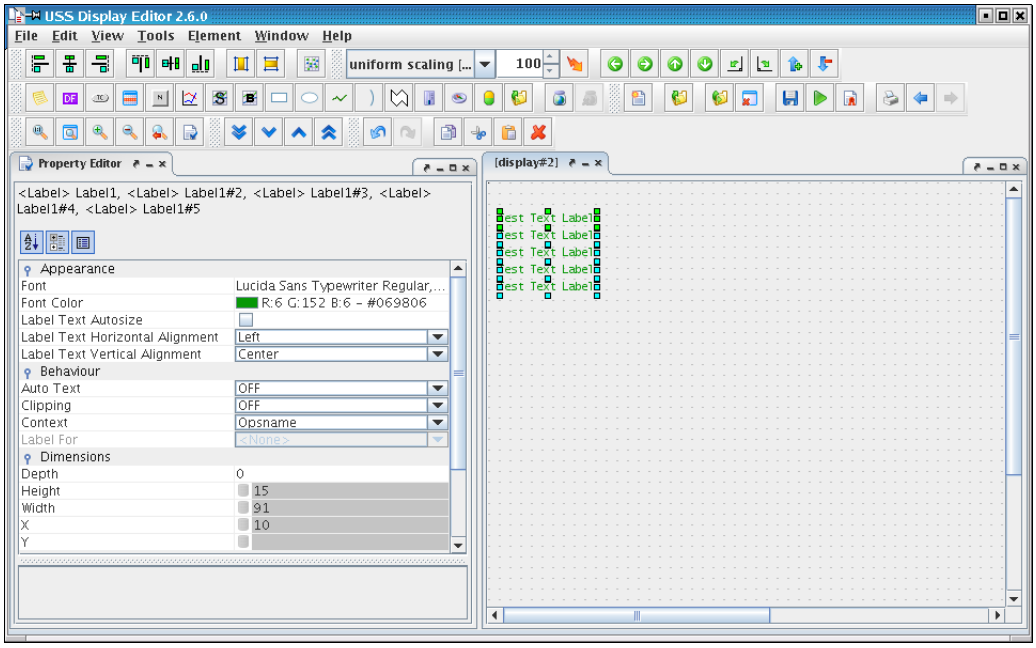
A text-label is created in the new display.

3. Make four more labels and select them all by using the keyboard combination: CTRL+A

4. In the Property Editor click to edit the colour and click '...' button to open colour dialog. Select tab: RGB and choose the green colour: Red: 6, Green: 152, Blue: 6

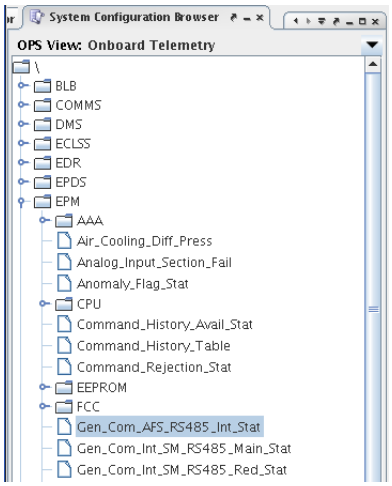


5. Click OK to the colour dialog.



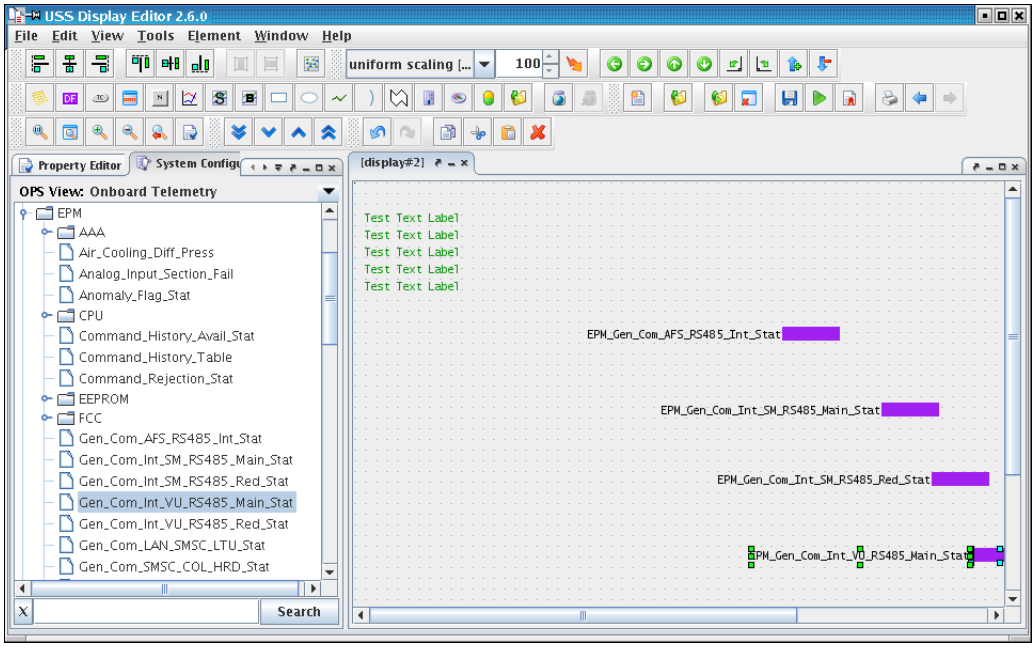
The colours of all the label fonts are changed to a dark green.

6. Open the System Configuration Browser and find in the OPS View for Onboard Telemetry, the parameter \EPM\Gen_Com_AFS_RS485_Int_Stat



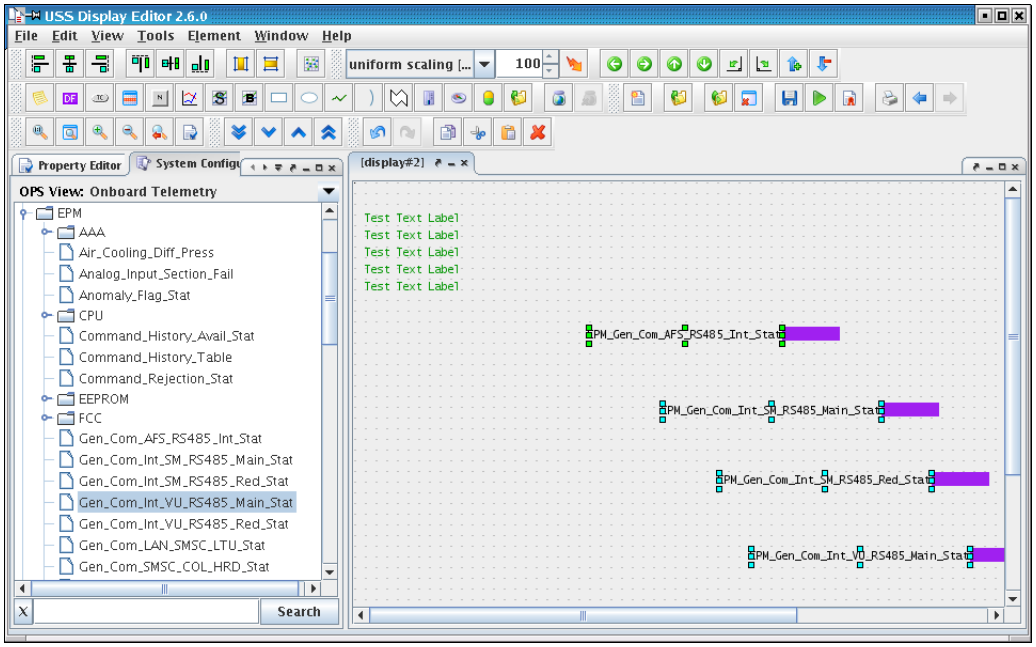
The end-item is selected.

7. Drag and Drop all the end-items to the display: Gen_Com_AFS_RS485_Int_Stat Gen_Com_Int_SM_RS485_Main_Stat Gen_Com_Int_SM_RS485_Red_Stat Gen_Com_Int_VU_RS485_Main_Stat



The end-items are added as four label/data-field pairs

8. Select the four labels by hold keyboard key: CTRL and left-click with mouse on labels.

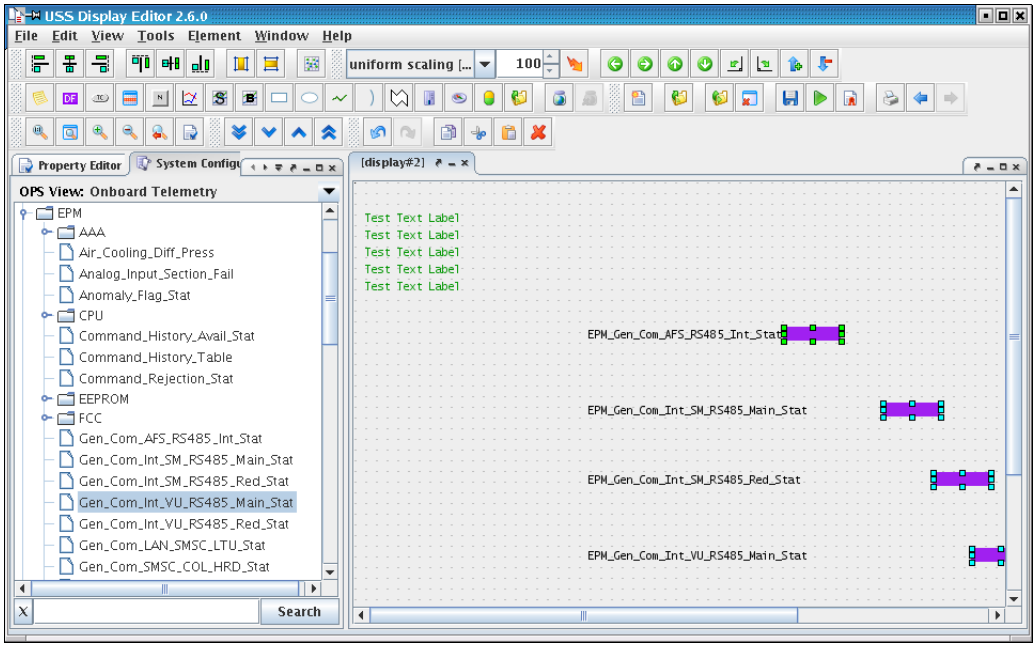


Four labels are selected.

9. Align the labels to their common left, by selecting form menu: Element | Align | Vertical Left

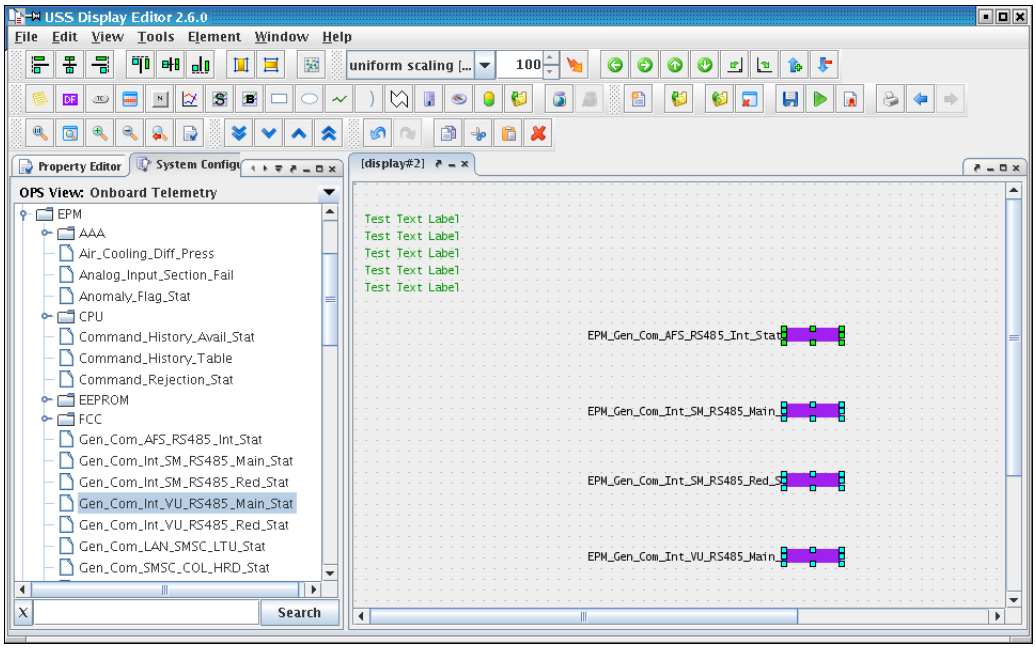
10. Deselect the labels by left-clicking with mouse on the display back-ground

11. Select the four fields by hold keyboard key: CTRL and left-click with mouse on labels.



Four fields are selected.

12. Align the fields to their common left, by selecting form menu: Element | Align | Vertical Left

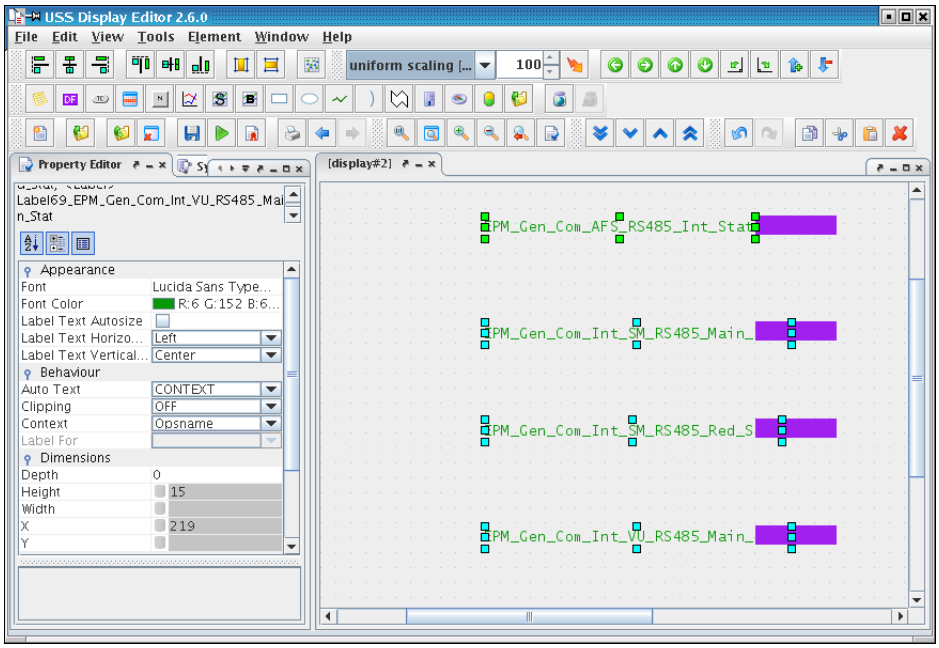


Four fields are aligned

13. Use the magnifying glass with a + in it, to zoom the display, to have a better look at the result

Display zoomed

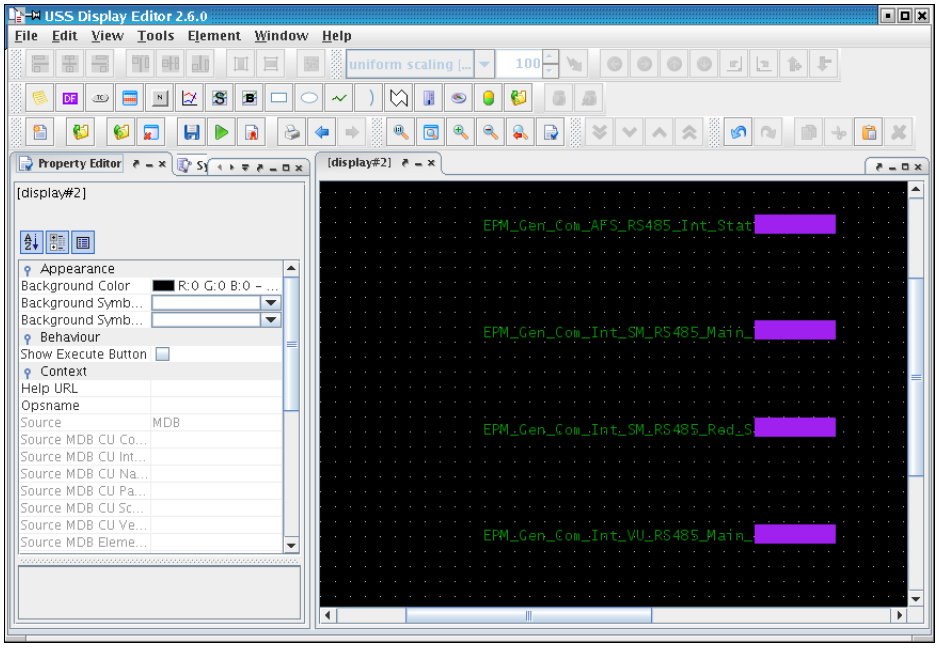
14. Select the four labels added from the System Configuration Browser and Open the Property Editor. In the Property Editor change the font colour to same as the previous labels.



Label colours change

15. Click on display back-ground (where there are no elements)

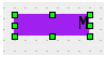
16. In the Property Editor select the Background colour of the display and change in to Black.



Display background is now black.

3.9.2 Lesson in working with ASCII displays (elements defaults and seperators)

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout
2. Add a field by selecting in the menu: Element | Add | Field



Field created in empty display

3. Make a standard field for use in ASCII displays, the field can be specialized. Change field properties to match the propertie sin the picture below.

Appearance	
Field Data Autowrap	☐
Field Data Formatting	
Field Data Text Horizontal Alignment	Left
Field Data Text Vertical Alignment	Top
Field DQI Override	☐
Field Overflow Behaviour	OVERWRITE
Font	Lucida Sans Typewriter Bold, 1, ...
Font Color	R:237 G:237 B:41 - #EDED29
Shape Fill Color	R:0 G:0 B:0 - #000000
Shape Fill Style	None
Show Data Indicators	☑
Show Unit	☐
Dimensions	
Depth	10
Field Data Columns	5
Field Data Decimals	0
Field Data Rows	1
Height	20
Width	70
X	10
Y	26
General	
Data Source	
Information	
Comment	
Name	Field1
Selected Elements	1 element
Tooltip	

Field properties after change

4. Make sure the field is selected, and use editor menu: Edit | Set as Element Default.
5. Try adding a new label by selecting in the menu: Element | Add | Field. Each field element created will get same apperance properties as the original field. To change the default, simply make a new standard field and select from menu: Edit | Set as Element Default

TIP

Before starting a larger display design, time can be saved, if default elements have been probably set

1. Continue with previous display.
2. Add a label by selecting in the menu: Element | Add | Label

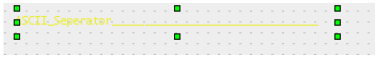
Label created in empty display

3. Set color of label font in property editor to RGB color: Red: 237, Green: 237 and Blue: 41
4. Double-click on the label to edit the label text

Label is in 'direct' label text edit-mode

5. Set label width to 300 pixels (in property editor)
6. Enter text: ASCII_Seperator_____

   i.e. "ASCII_Seperator" followed by 30 underscores



Label is used as seperator

7. Set display background to black (click on display background and edit the background color)
8. Set display width to 320 pixels and height to 100 pixels

Label is used as seperator

TIP

ASCII displays contains no graphical elements, but labels can be used in some cases to mimic seperators etc.

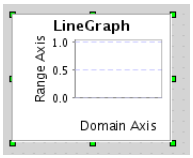
3.10 Making a Graph Display

The USS Editor can be used to create displays containing Graph of various kinds in the USS display format. Graphs can be used to show larger data-sets in a more intuitive way.

3.10.1 Lesson in creating Graph displays

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout

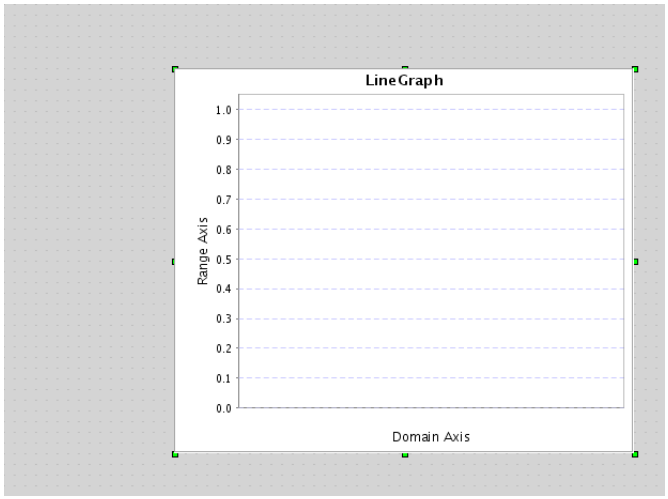
2. Make a Line Graph by selecting from menu: Element | Add | Line Graph



A empty Line Graph is made in the upper left of the new display.

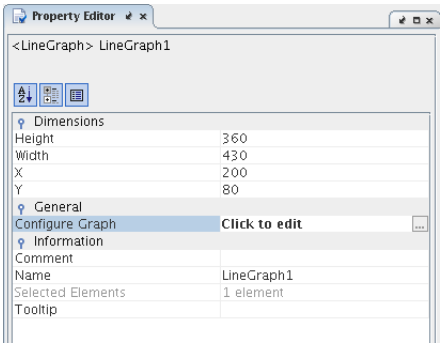
3. Use the mouse and left-click and drag on the right-bottom pick-control point (green square in bottom-right of line graph). This will resize the graph.

4. Use the mouse and left-click (and hold) in the middle of the graph and move the graph to the middle of the display.

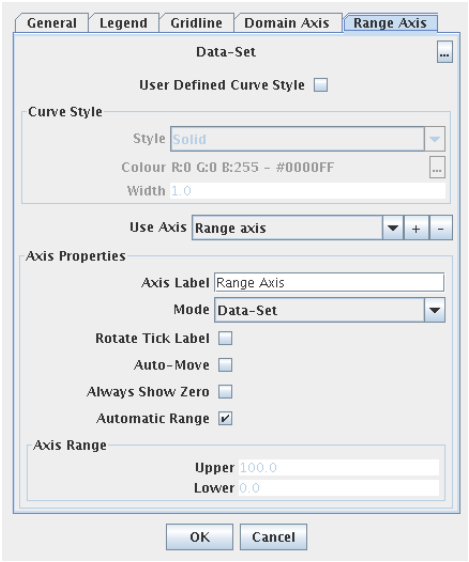


The Line Graph is moved and resized

5. Open the Property Editor to see the graph properties.

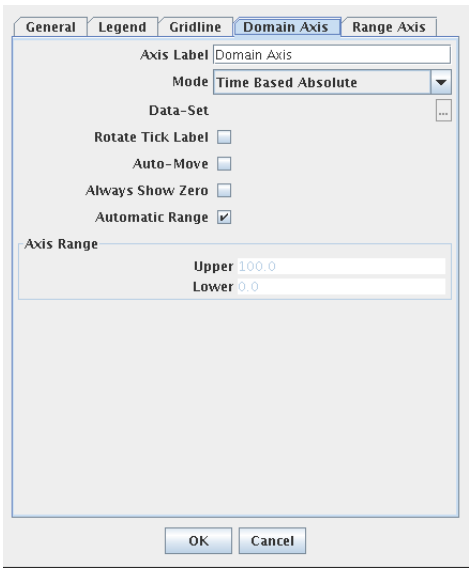


6. Click on ...-button in property: Configure Graph.



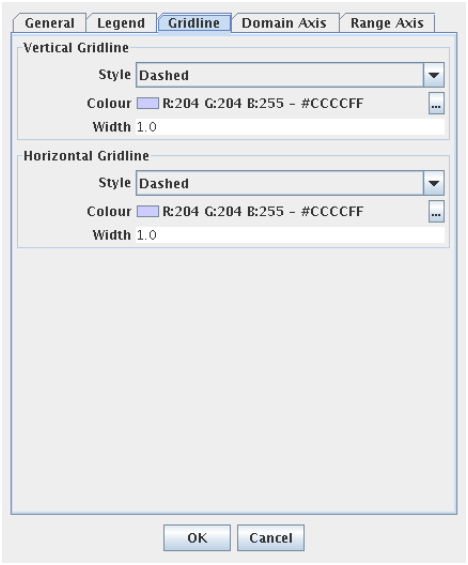
The Graph Dialog opens for the Line Graph

7. Click on domain tab



The Graph Dialog shows the domain tab for the Line Graph

8. Click on grid line tab

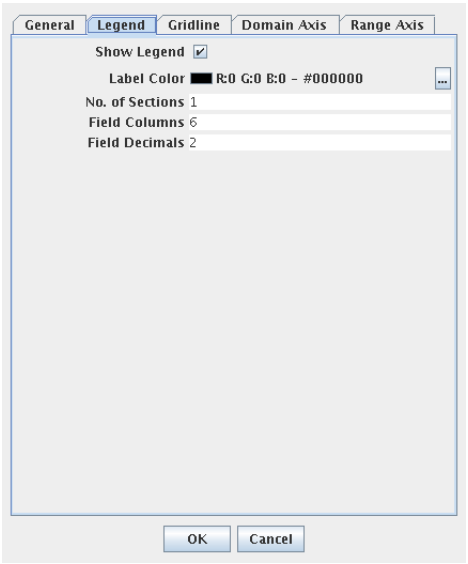


The Graph Dialog shows the grid line properties tab for the Line Graph

9. Change the horizontal and vertical grid colours to Brown from My Colours and click OK-button

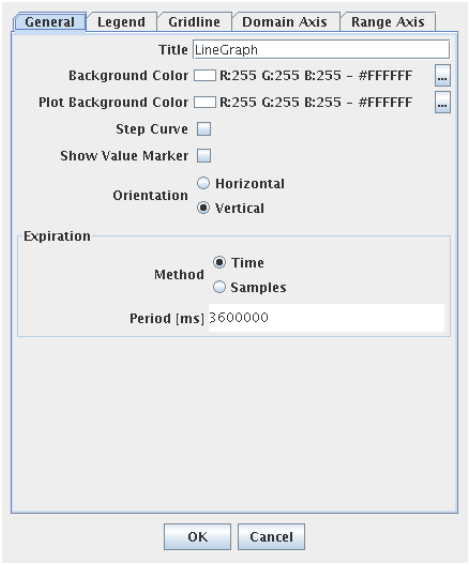
Grid colour are updated

10. Click on legend tab



The Graph Dialog shows the legend properties tab for the Line Graph

11. Click on general tab

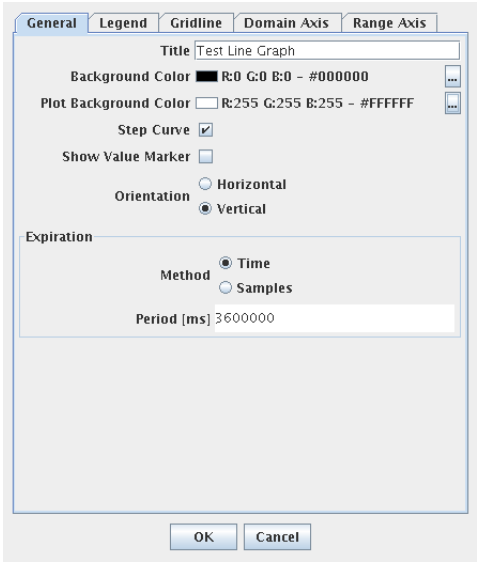


The Graph Dialog shows the general properties tab for the Line Graph

12. In the general tab click the ...-button to open the Background colour selection dialog and select under My Colors the colour black

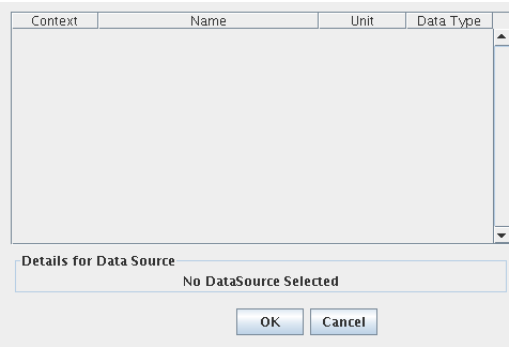
The black colour under My Colours

13. Click OK to the colour dialog



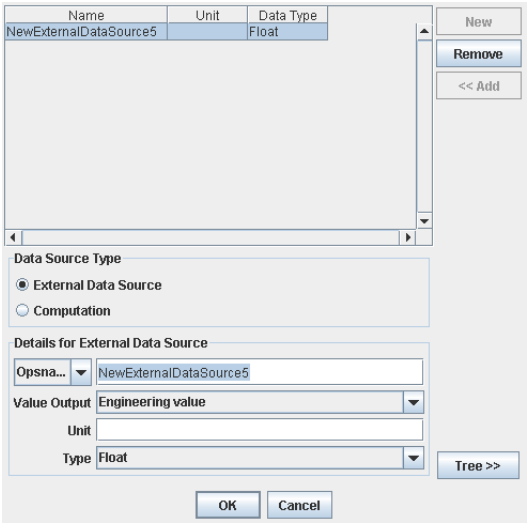
Background colour is now set to be used in the Line Graph

14. Go to the range-tab and click the ...-button for the property: Data-Set



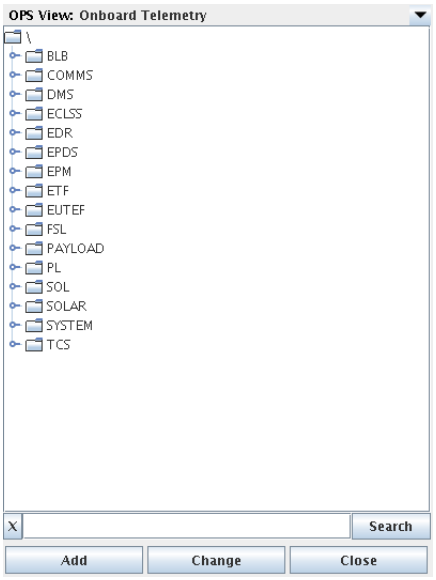
The Data Source Dialog is open. The dialog is used in the editor for all data source configuration for elements

15. Click the Add-button to add a new data source

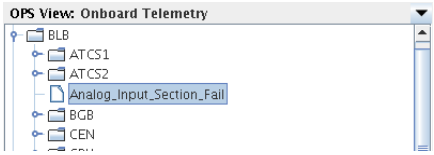


The Data Source Dialog add a new external data source

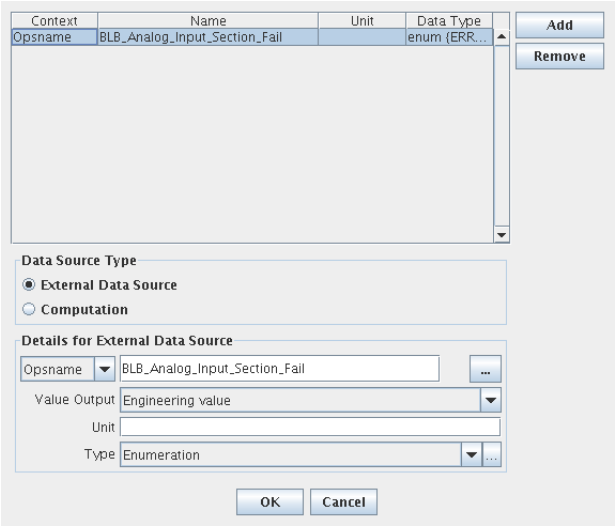
16. Under the details for the External Data Source click the ...-button to open the System Configuration Browser.



17. In the System Configuration Browser browse to the following: \BLB\Analog_Input_Section_Fail and select it

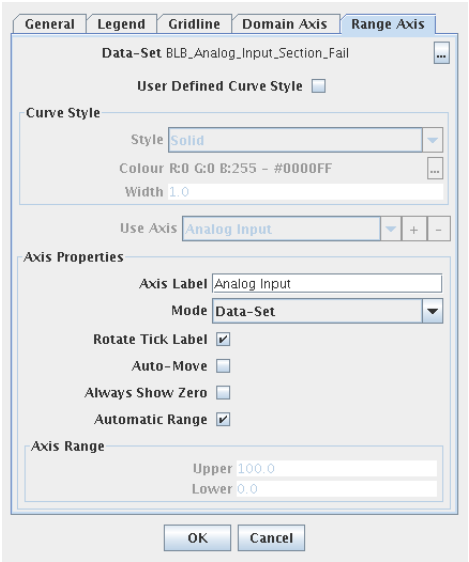


18. Click the Change-button followed by the Close-button



The External Data Source is now updated with the SCOE information

19. Click the OK-button to activate the data-source editing



The External Data Source is updated in the Graph dialog

20. Click the OK-button to activate the graph editing

The graph in the display is now updated to match the configuration

21. To Preview the graph: Select from menu: Tools | Previewer

The line graph is previewed

3.10.2 Lesson in creating Strip-Graph displays

1. Start by opening the editor via installed icon Screenshot of USS Editor started in default layout

2. Make a Line Graph by selecting from menu: Element | Add | Strip Graph

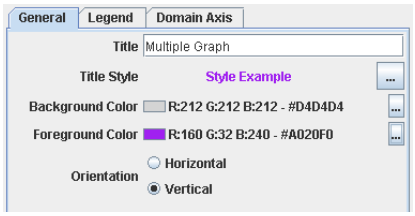
3. Change width to 500 pixels and height to 500 pixels by changing the properties in the property editor

4. Find the specific strip-graph properties in the property editor

Specific strip graph properties

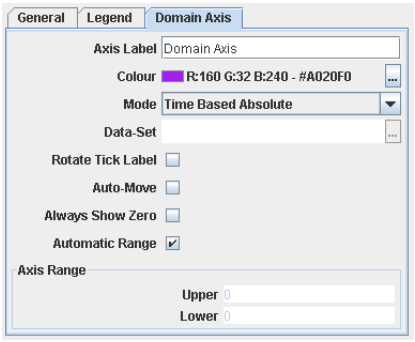
5. Add a new sub-graph (line-graph inside the stripgraph element) by clicking the button: + (plus) in the property: Select Sub Graph.

6. Click on button: ... in property: Configure Graph and set the title, foreground and bacground colors to match picture below.



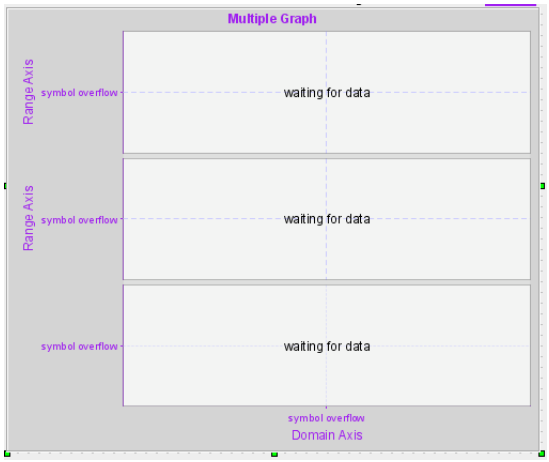
Strip graph properties in graph configuration dialog

7. Click on domain tab



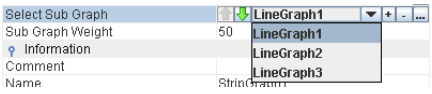
The Graph Dialog shows the domain tab for the Strip Graph

8. Actualize the graph dialog changes by clicking OK-button



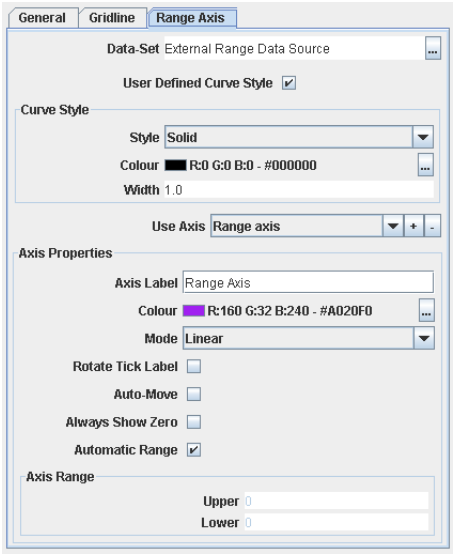
Strip graph propeties changed, notice background of stripgraph changed, but sub-graph still have their own background preserved.

9. Click and select 'Line Graph 2' in property editor



The sub-graph order, selection etc. can be changed via the property: Select Sub Graph

10. Open the sub-graph edit dialog by clicking button: ... in property: Select Sub Graph



The sub-graph specific properties, shown range axis properties

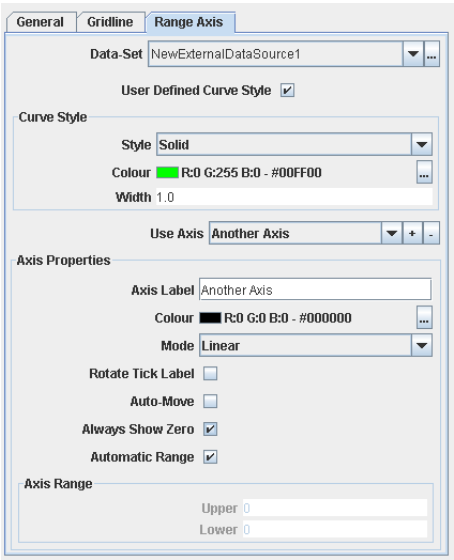
11. Add default datasource by clicking the button: ... in the property: Data-Set. Twice click button: New in datasource dialog

12. Change the datasources in the datasource dialog to type: Integer

13. Add another datasource

14. Click datasource dialog button: Ok

15. The newest aded datasource is now selected, add a new axis for that datasource by clicking button: + in property: Use Axis Set title in Axis properties to: Another Axis



Second datasource uses a second axis

16. Click sub-graph dialog button: Ok

17. Do the same for the other sub-graphs (select sub-graph and edit adding two datasources to to range axis)

18. Save display to default name and location (menu: File | Save)

19. Preview Strip graph display in the editor previewer (menu: Tools | Previw Display...)

TIP

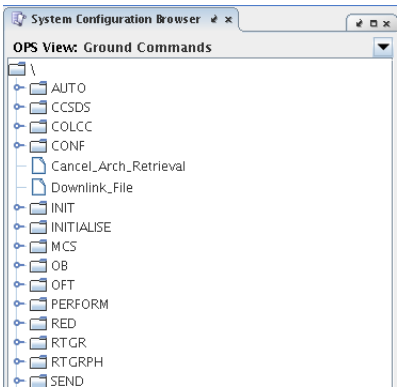
Multi-axis graphs must have datasources with output numeric values (e.g. type: Integer, Float)

3.11 Create Commanding Display

The USS Editor can be used to create commanding displays in the USS display format.

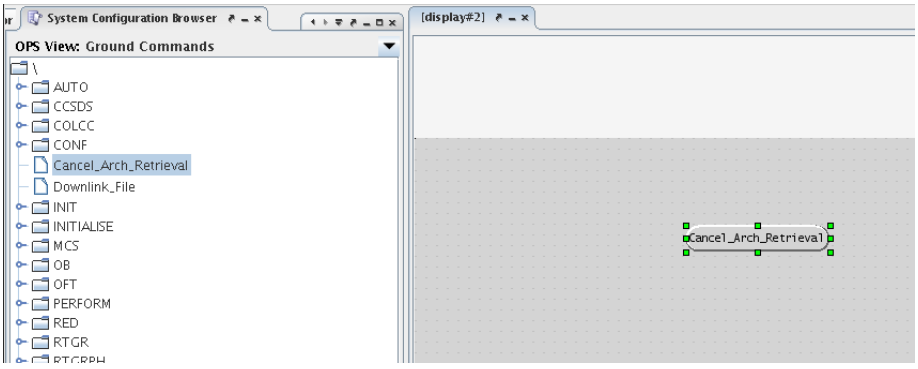
3.11.1 Lesson in creating Commanding displays

1. Start by opening the editor via installed icon. Screenshot of USS Editor started in default layout
2. Open the System Configuration Browser.



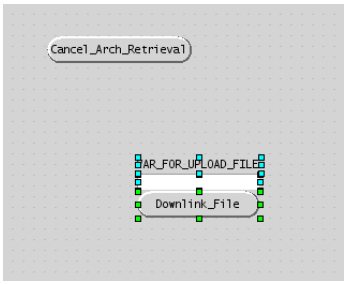
The System Configuration Browser is shown

3. Shift the System Configuration Browser to OPS View Ground Commands, using the view selector on the right.
4. Browse to the End-item \Cancel_Arch_Retrieval and select it.
5. Use the mouse and left-click (and hold) on the end-item and drag it to the open display. Select a String Command from the popup.

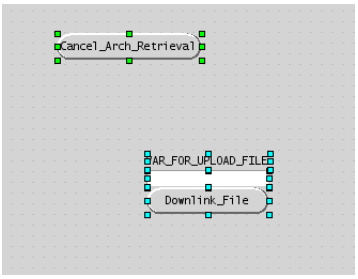


A structured telecommand button is made in the display with text Cancel_Arch_Retrieval

6. Browse to the End-item \Downlink_File and select it. Select a Structured Command from the popup.
7. Use the mouse and left-click (and hold) on the end-item and drag it to the open display.

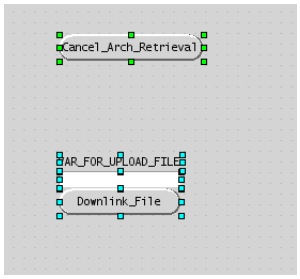


8. Use the keyboard key combination: CTRL + A , to select all.



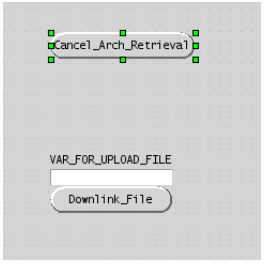
All created display elements from the drag-and-drop operations are selected

9. Choose from menu: Element | Align | Vertical Left



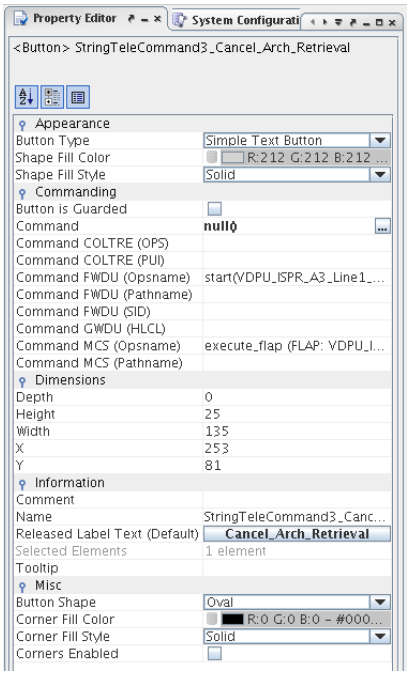
Elements are aligned to the left-most position

10. Left-click to select the Cancel_Arch_Retrieval Telecommand (top-most).



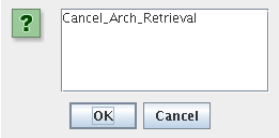
Telecommand selected

11. Open the Property Editor.



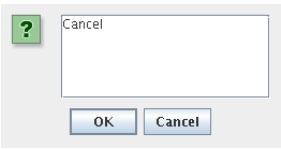
Properties of Telecommand is shown

12. Click in the Property Editor on the button with text: Cancel_Arch_Retrieval for the property Released Label Text.



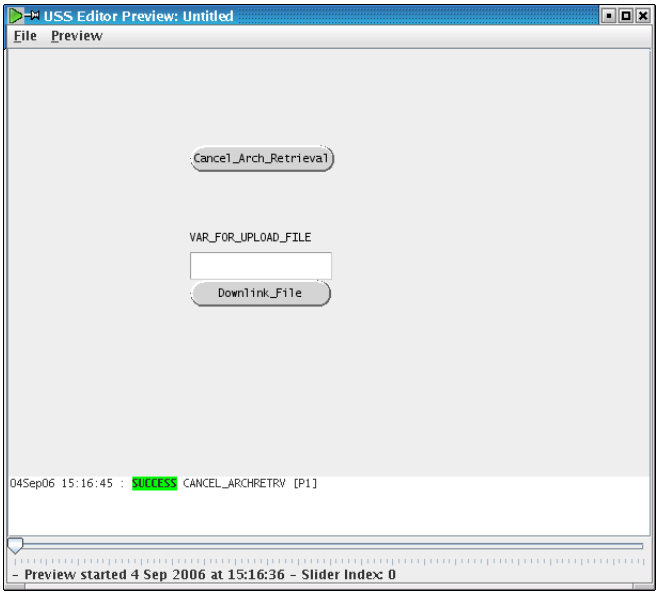
Dialog with changeable text is shown

13. Change text to: Cancel



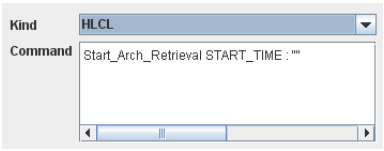
14. Click OK to change property.

15. Preview the created display by choosing: Tools | Preview (opens Preview Frame), click the created telecommand with label: Cancel



Previewer shows simulated command execution at the bottom

16. Close Preview.
17. Select command in System Configuration Browser: Start_Arch_Retrival.
18. And drag it to the display.
19. Select String command from the popup.
20. Click the button: ... in the property: Command in the property editor.



String Command Editing

21. Change the string command from: Start_Arch_Retrieval START_TIME : "" to Start_Arch_Retrieval START_TIME : "10"
22. Click Close to commit editing. The Command argument is now chnaged.
23. Select command in System Configuration Browser: Start_Arch_Retrival again, drag it to display and select a structued command instead.
24. Select the newly added button and click the button: ... in the property: Command in the property editor.

Structured Command Editing

25. Select the parameter: START_TIME and change the value from nothing to 10.
26. In the command preview field below, the result can be seen (same as with string command editing).
27. Click Close to commit editing. The Command argument is now changed.

TIP

In the Property Editor, the property: Command contains a button: '...', which gives access to the command editing dialog

TIP

In the Property Editor, the property: Command contains a button: 'X', which gives deletes the command (after this '...' will give the possibility to change command type)

TIP

USS preferred / default command type is structured command, the structured command gives a less error-prone method for contruction telecommands

TIP

Structured commands can be referenced / combined with input fields and comboboxes to make advanced commanding

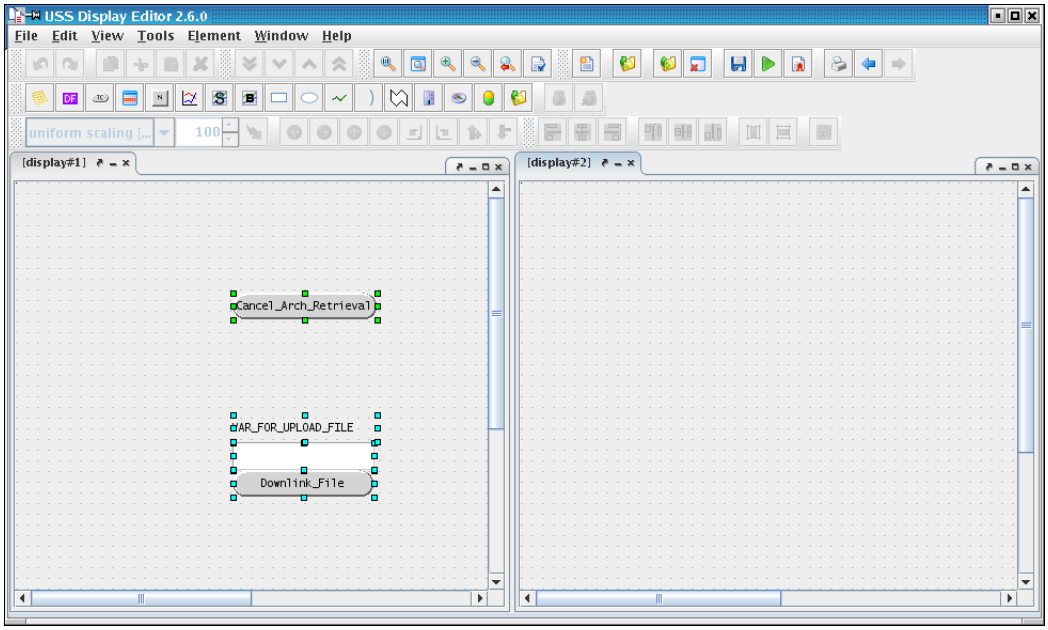
3.12 Navigation Display

The USS Editor can be used to create displays that can be navigated by navigation-buttons in the USS display format. The navigation-buttons are complex hyperlinks between displays, using their relative path (navigation cannot happen on the display itself). Pictures in this lesson show the result from previous lesson as one of the displays used.

3.12.1 Lesson in creating Navigation displays

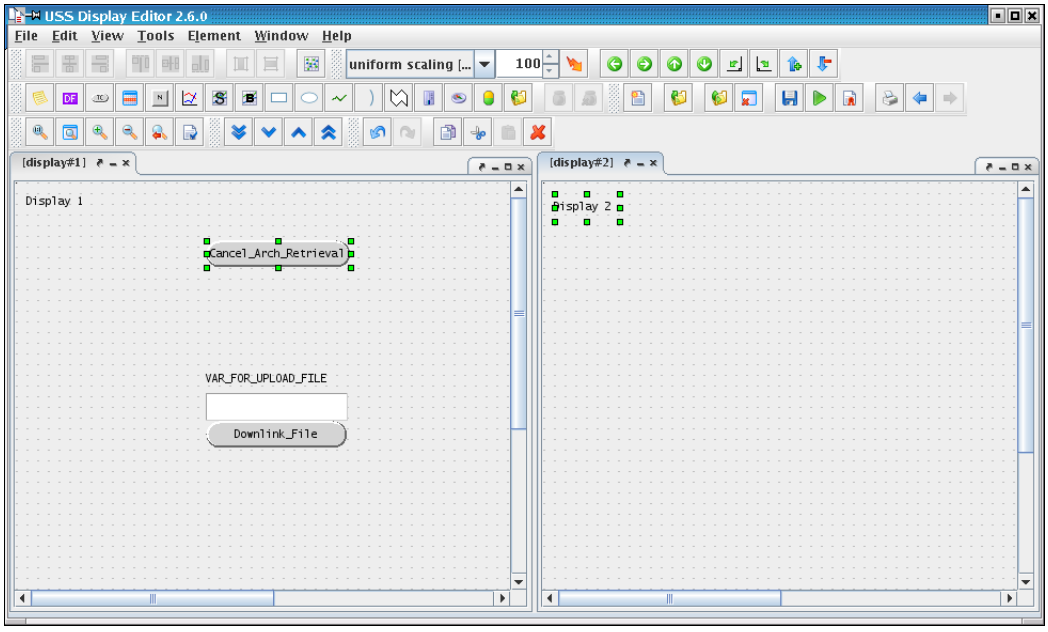
1. Start by opening the editor via installed icon, close all tool-views and make a new display, so that the editor contains two new displays. Screenshot of USS Editor started in default layout

2. Arrange the displays by dragging them (left-click-hold on display-tab), so that both are visible at the same time.



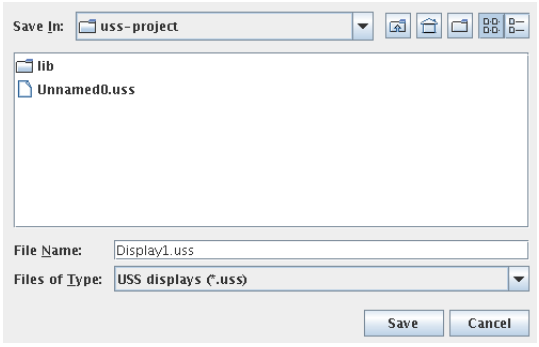
3. Make a label in each display. - Select display by left-clicking on display-background - Make a label from menu: Element | Add | Label

4. Change label-text in first display to Display 1 and Display 2 for label-text in second display. - Double left-click on label to start in line editing of label.



5. Click on Display 1 background to select it.

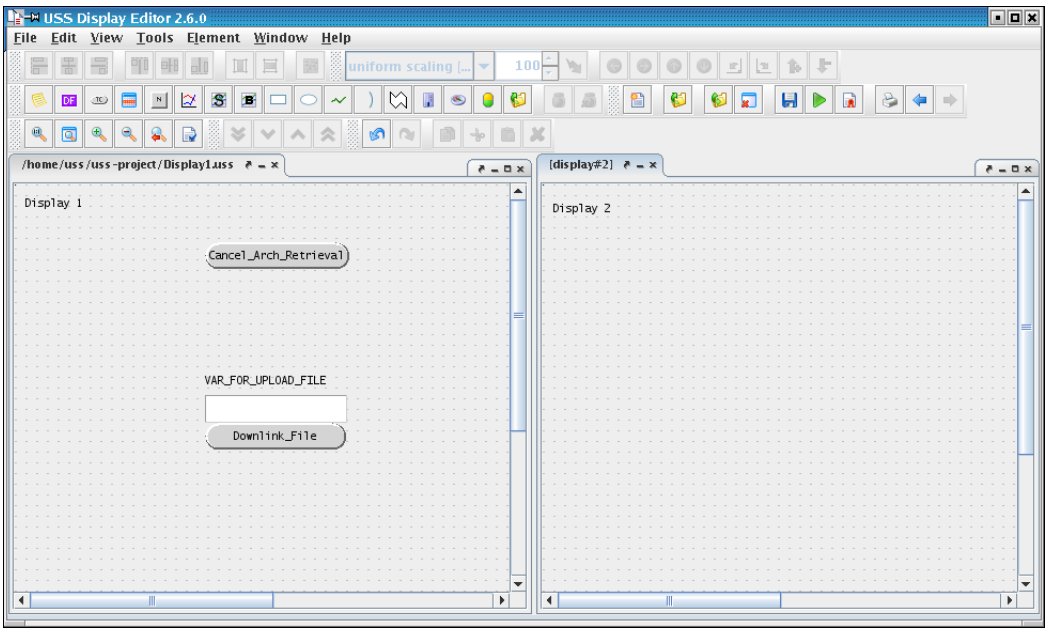
6. Save display by selecting from menu: 'File | Save'. Type in file name: 'Display1.uss'



Save dialog is shown

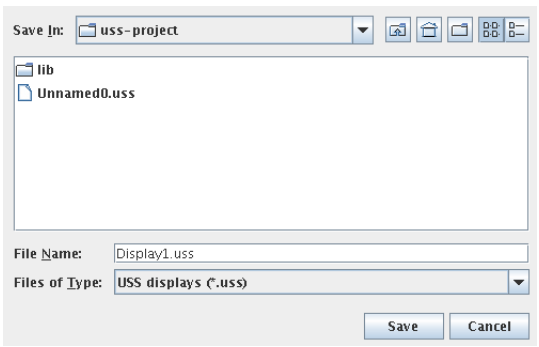
7. Click OK to save dialog to save display.

8. Use the mouse and left-click on the second display background to select it.



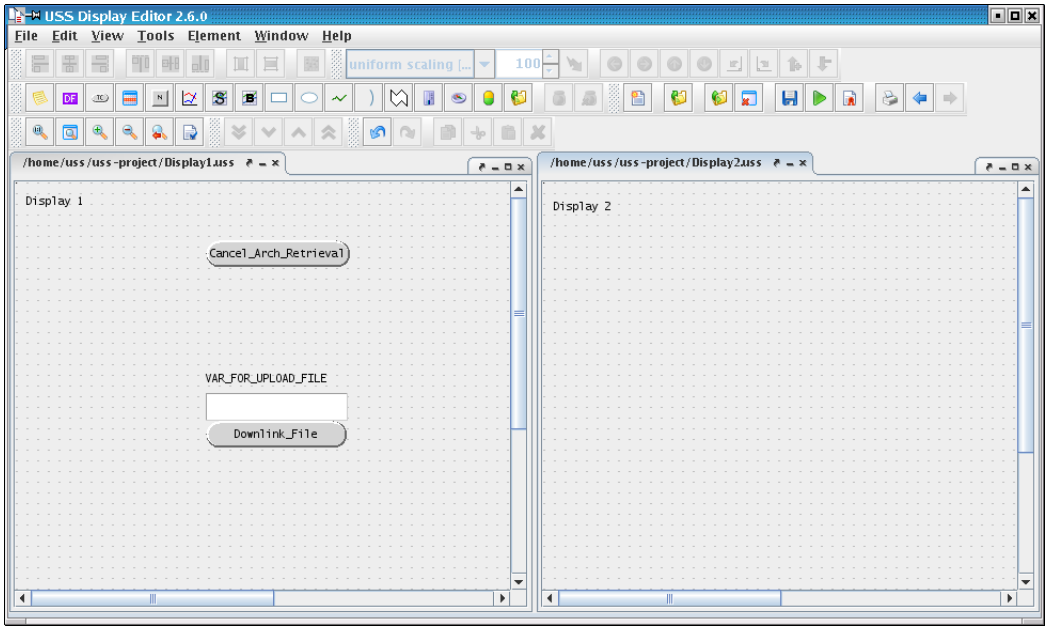
Second display is selected

9. Save display by selecting from menu: File | Save . Type in file name: Display2.uss



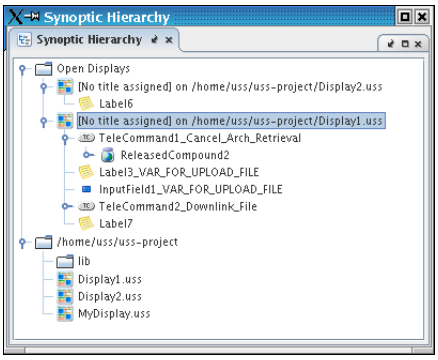
Save dialog is shown

10. Click OK to save dialog to save display the second display



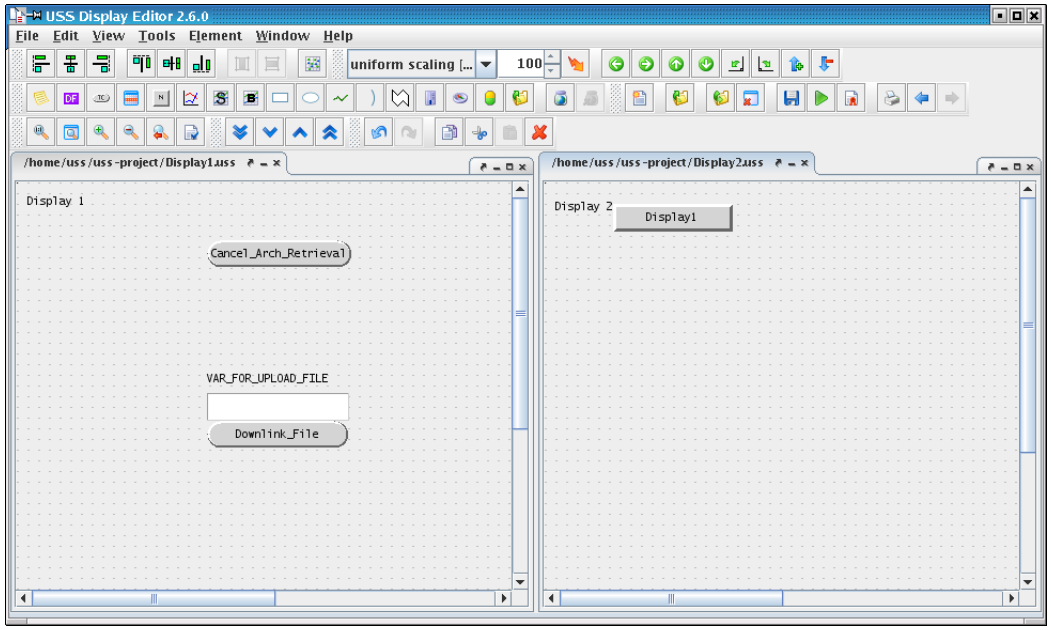
Second display saved

11. Open tool-view: Synoptic Hierarchy



In the top of the Synoptic Hierarchy (shown un-docked) the two display with their respective label are shown. In the bottom the project-folder is shown, containing the two saved displays

12. Right-click on Display1.uss in the bottom part of the Synoptic Hierarchy and select from pop-up menu: Add Navigation to Display

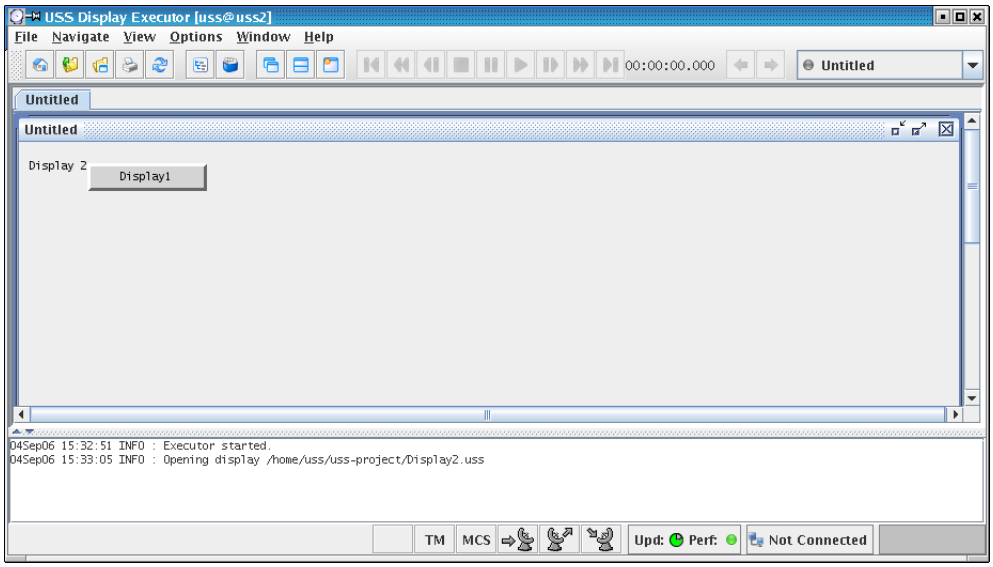


A navigation button is added to currently selected display (Display2.uss). The navigation button will navigate (upon activation) to Display1.uss, when Display2.uss is executed

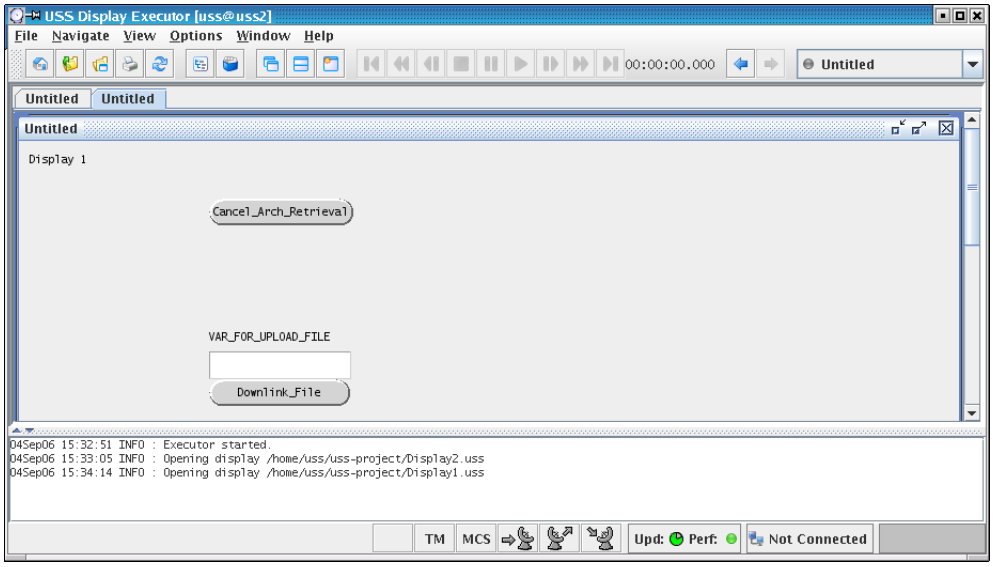
13. Save both displays again and close the Editor

14. Open the Executor from installed icon

15. In Executor open display file: Display2.uss from the Project-folder



16. Click on the navigation-button with text: Display1



Executor opens Display1.uss in a new tab

17. Close the executor and open the editor again
18. Make a new display and save it together with the two previous, naming it: Display3.uss
19. Make a navigation button and set the button text via the property editor to: Change to Display

New Navigation button

20. Locate the greyed icon for configurating the dynamic property: Navigation Target

Navigation Properties of the navigation button

21. Click the greyed icon, wait for the datasource dialog to open
22. Make new datasource and select computation, and enter the expression: if VTC2_Redun_Temp_VTC > 10 then "Display1.uss" else "Display2.uss"

Navigation Expression for the navigation button

23. Save the new display again
24. Preview the display and use slider at the bottom to change which display will be opened by navigation button

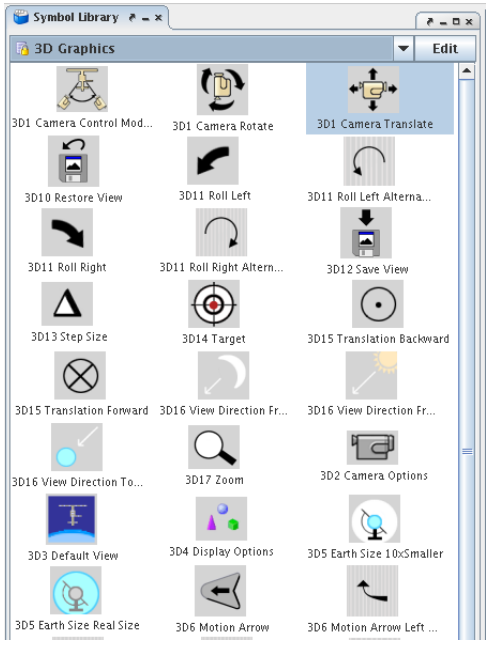
3.13 Add Symbols

The USS Editor can be used to create symbols that can be reused in the displays. Symbols are contained in Symbol Libraries.

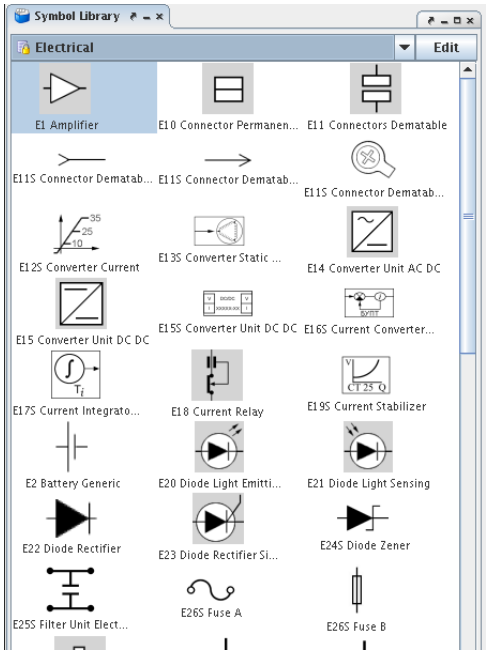
3.13.1 Lesson in creating Symbols

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

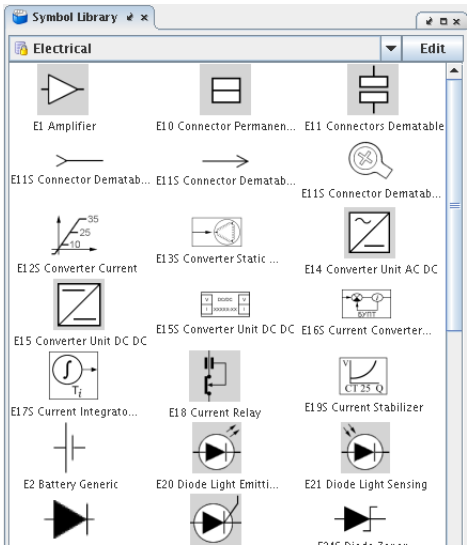
2. Open the Symbol Library from menu: View | Symbol Library



3. In the Symbol Library select from the drop-down box (ComboBox) the symbol library: Electrical

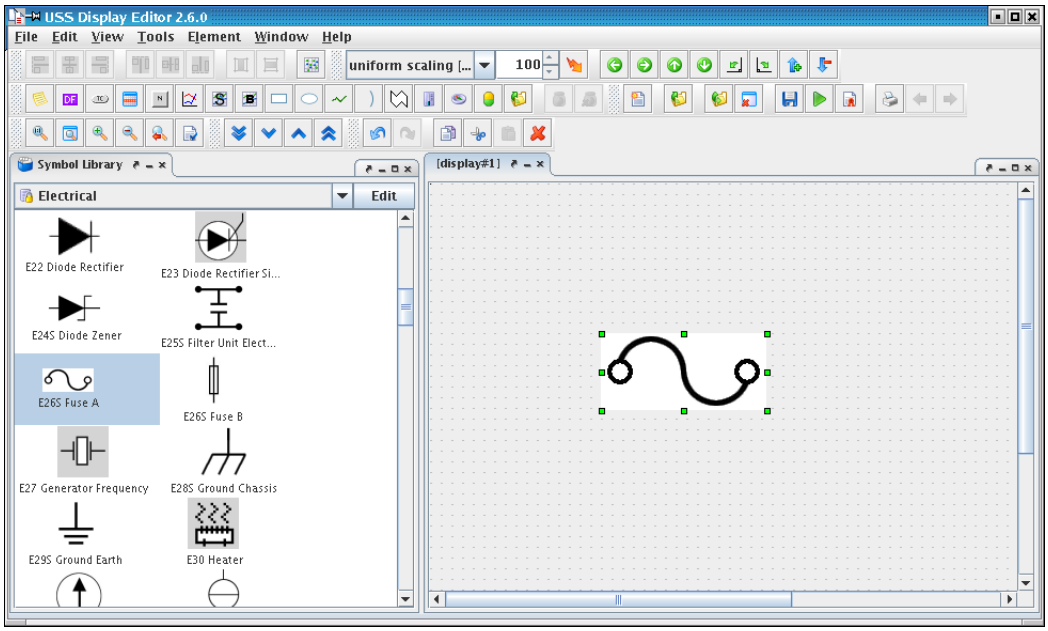


4. In the library: Electrical select the symbol: E265_Fuse, by left-clicking with the mouse on the symbol.



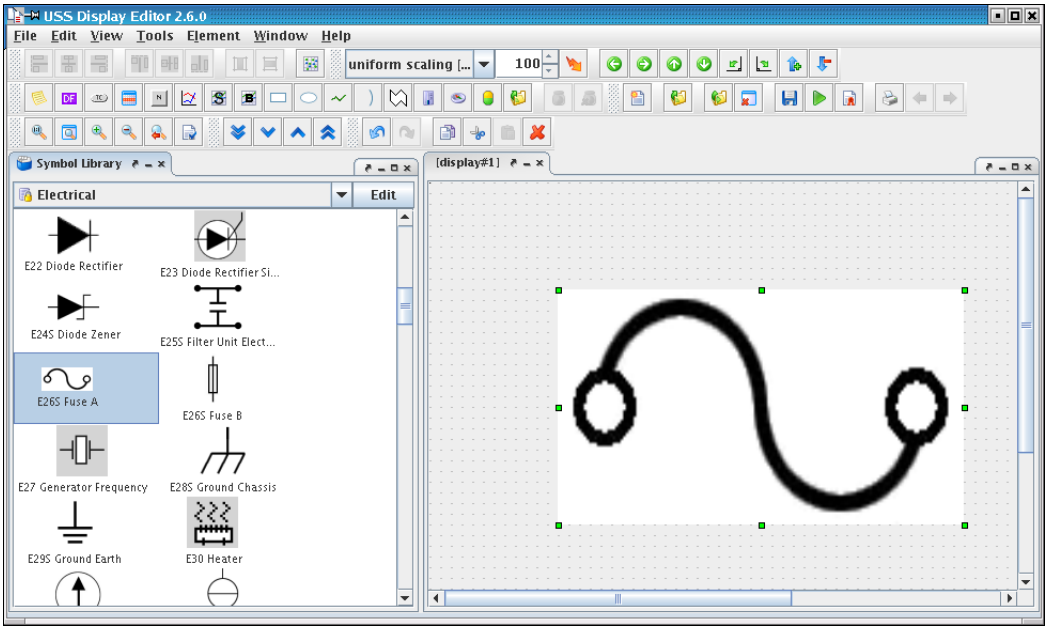
Symbol E265_Fuse selected in Symbol Library (shown un-docked).

5. Left-click (Hold) and drag the symbol to the middle off open display, to add it to the new display.



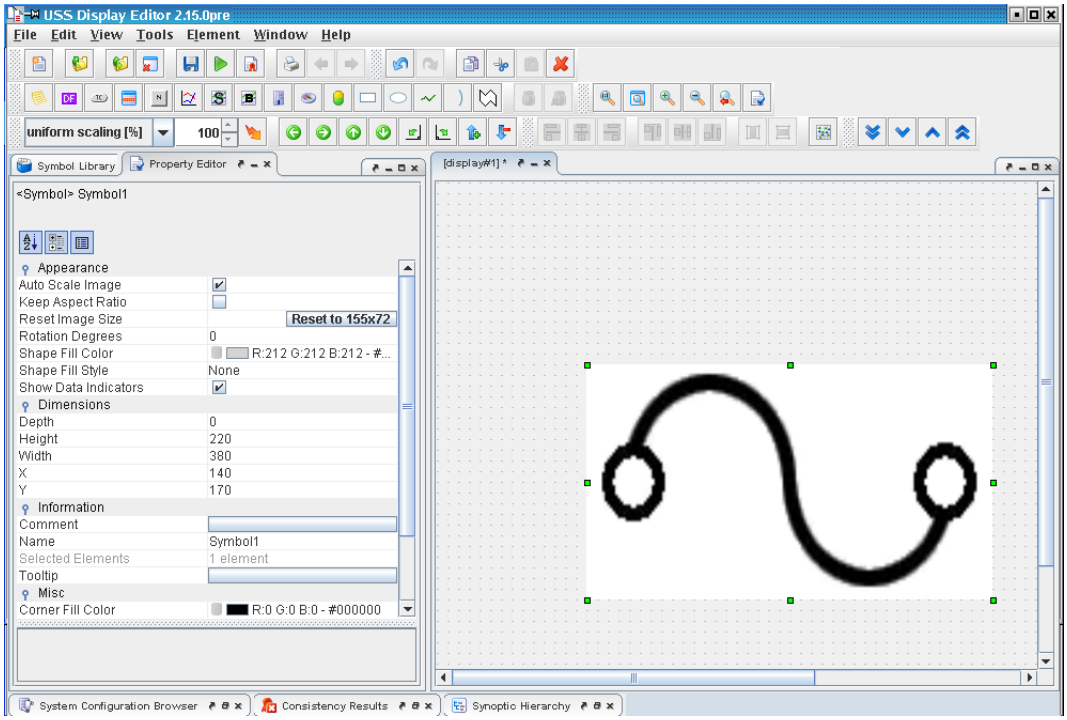
The Symbol is added to the display, as an element using the image material given by the symbol. This element contains a reference to the symbol, so that if the symbol library is updated, the symbol-element in the display, will be likewise.

6. Resize the symbol-element by dragging its control-points in the corners (make it larger).



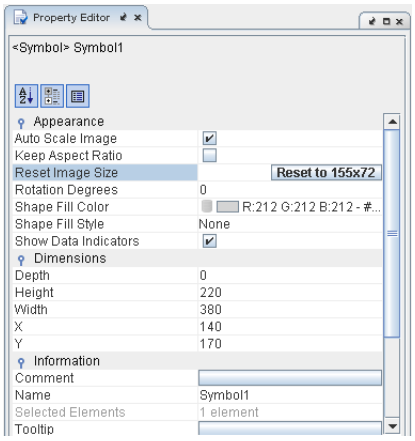
The symbol-element is resized.

7. Open the tool-view: Property Editor to show the properties for the element.



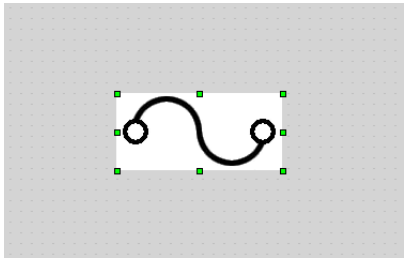
Symbol-element properties are shown.

8. Undock the Property Editor and click the property: Reset Image Size - button to reset back to the original size of the symbol.



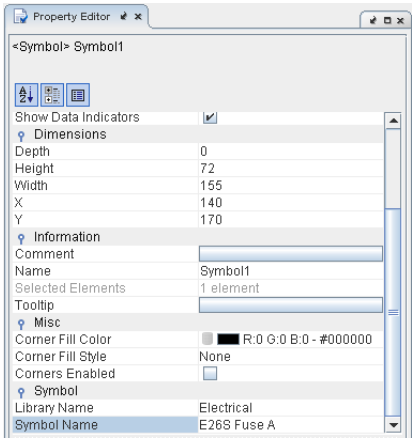
Property is reset.

9. Re-dock the Property Editor.

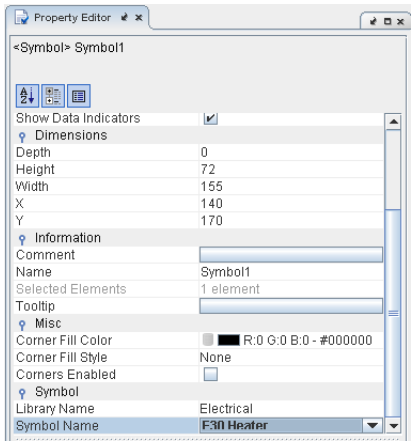


The display shows the reset symbol-element in the display.

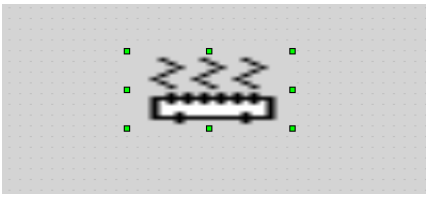
10. In the property editor click the property: Symbol Name.



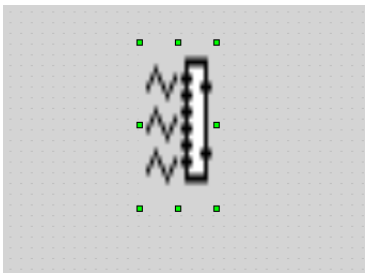
11. In the property editor change the property: Symbol Name, to E30_Heater.



12. Observe the the display symbol-elements icon is changed to the heater icon.

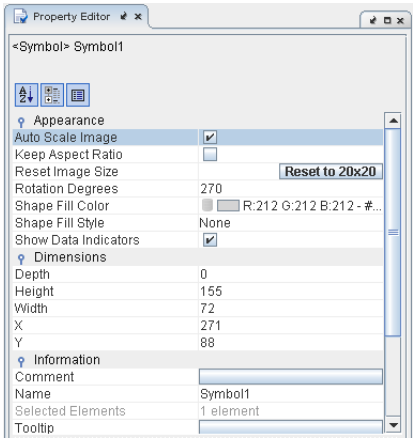


13. Rotate the symbol-element by selecting from menu: Element | Element | Rotate | Left

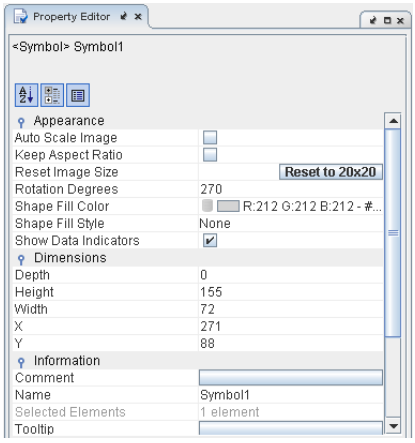


The symbol-element is rotate counter-clockwise (left). This feature is available not only for symbol-elements. Some complex elements, cannot be rotated.

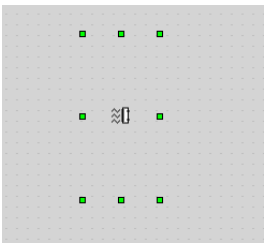
14. In the property editor click to the property: Auto Scale Image.



15. Uncheck the property: Auto Scale Image.

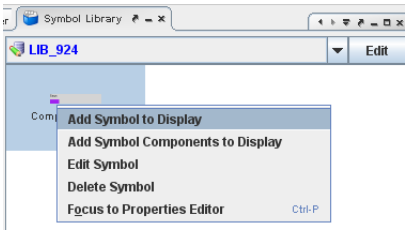


16. Observe that the icon of the symbol-element is resize to its real size (in pixels) and but the extend of the symbol-element is persistent.



TIP

When adding compound symbol to a display, this can be done in two distinct ways: As a symbol reference (classical way), and as a the components inside the compound symbol (recreating the components)



Popup menu in symbol library view

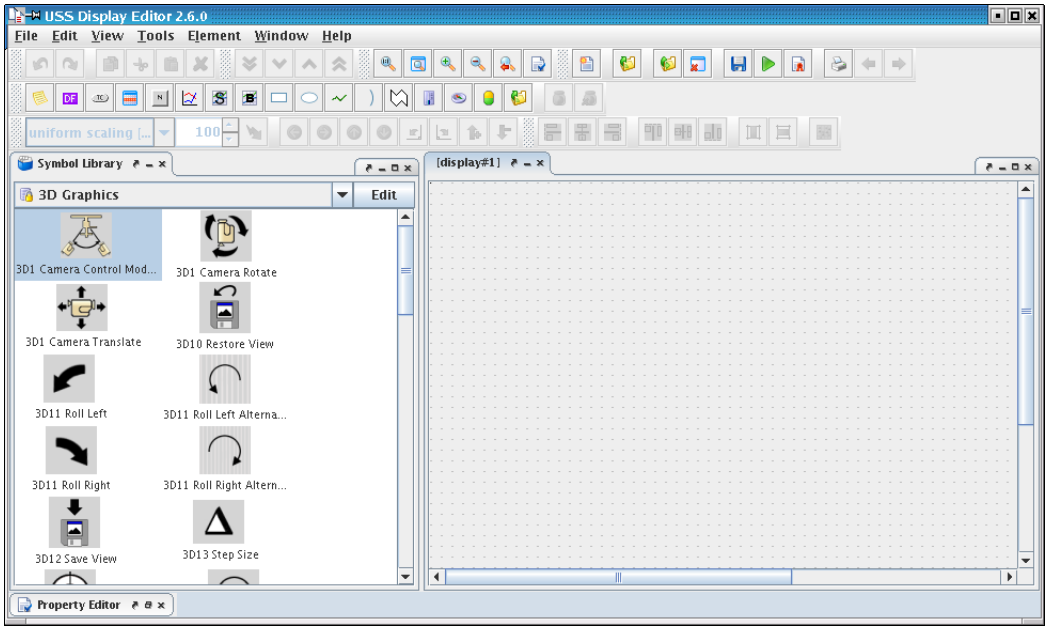
3.14 Create New Symbols

The USS Editor can be used to create new symbols and symbol-libraries.

3.14.1 Lesson in creating Symbols

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

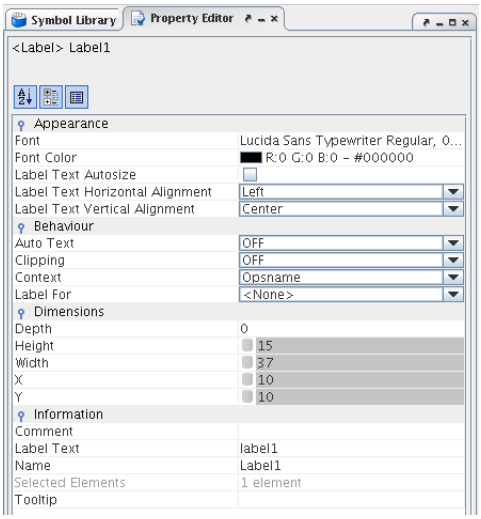
2. Open the Symbol Library from menu: View | Symbol Library



3. Make a new label by selecting from menu: Element | Add | Label

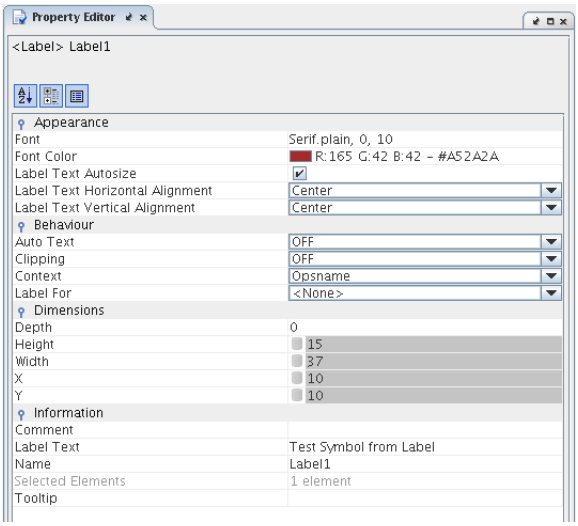
Label is created in the display.

4. Open the property editor from menu: View | Property Editor



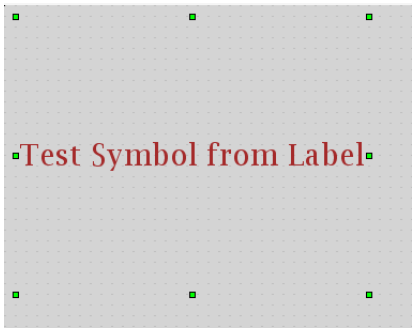
Properties are shown from label.

5. Set the properties to match approximately, following picture:



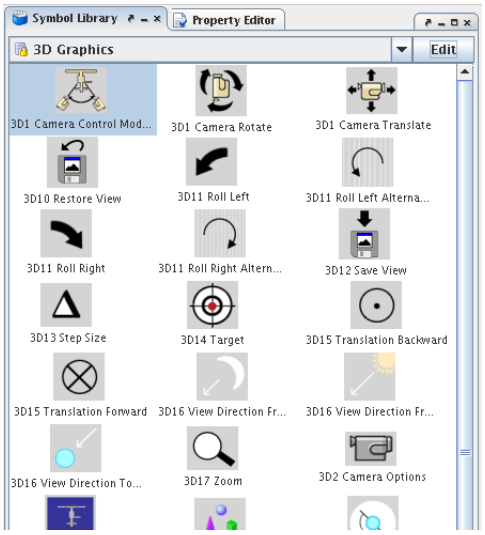
Change: Font, Font Color, Label Text Autosize, Label Text Horizontal Alignment, Label Text Vertical Alignment and Label Text. Giving the following label result:

6. Resize the label by dragging its control-points in the corners (make it larger).



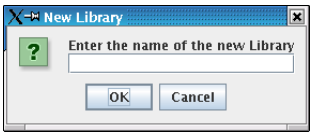
The label is enlarged; text is followed because we set the property 'Label Text Autosize' to checked.

7. Open the Symbol Library from menu: View | Symbol Library.



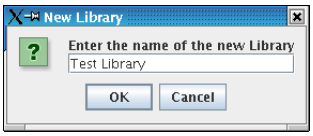
Symbol Library shown.

8. Click the button: Edit, and in the pop-up menu select: Create New Library

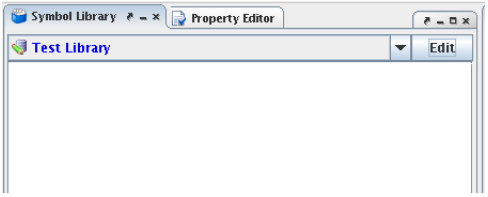


Dialog for setting new library name is shown.

9. Set name to: Test Library.

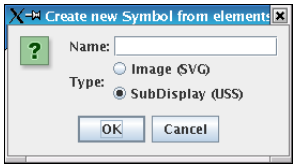


10. Click OK, and select new Library in the drop-down-box (ComboBox)



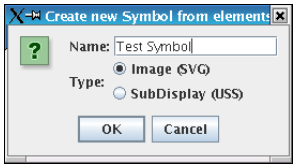
A new empty symbol library is created.

11. Add a new symbol, created from the previously created label, by clicking the button: Edit in the symbol library and selecting: Create Symbol from Selected Elements

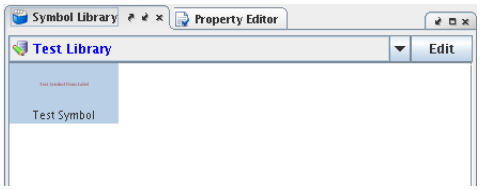


Dialog shows the configuration possibilities for the new symbol.

12. Select Type: Image (SVG) and name the symbol: Test Symbol.

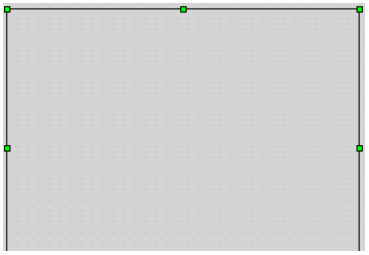


13. Click button: OK to create symbol.



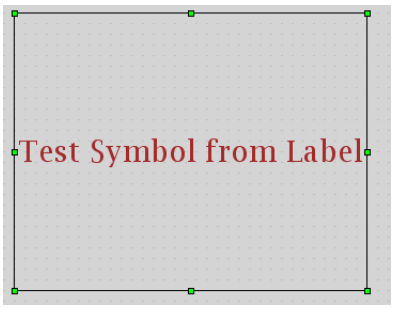
The symbol is created in the new library. Symbols can only be created in the user symbol libraries, because predefined symbol libraries are distributed as part of the USS package an will be overwritten on system update. They are USS-version persistent so to speak.

14. Now add a rectangle to the display from menu: Element | Add | Rectangle.

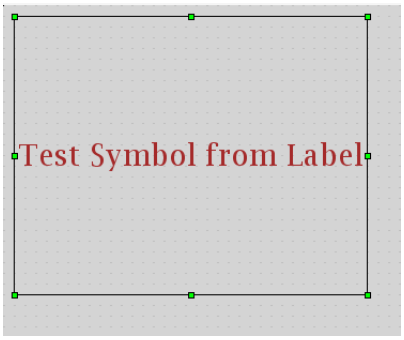


A rectangle is added to the open display.

15. Move the rectangle, so that it acts as a border to the previously created label.

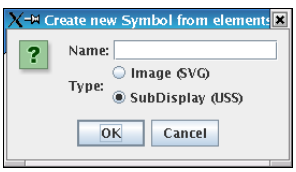


16. Select both by using the menu: Edit | Select All.



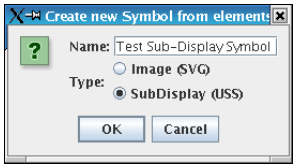
Label and rectangle are selected.

17. In the symbol library press button: Edit, and select from pop-up menu: Create new Symbol from selected Elements.

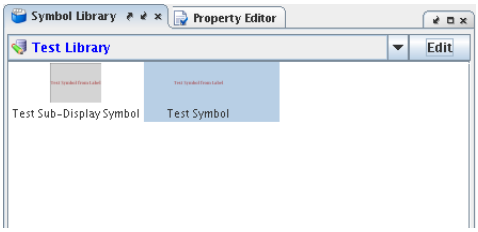


Configuration dialog is shown.

18. Set the properties, Name: Test Sub-Display Symbol Type: SubDisplay (USS)

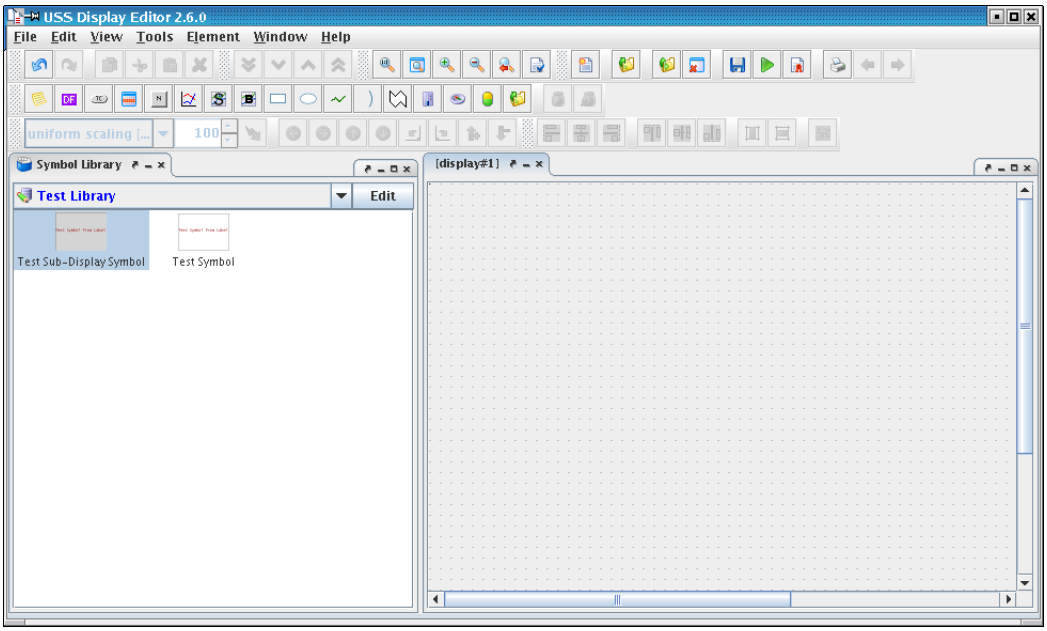


19. Click button: OK to create symbol.



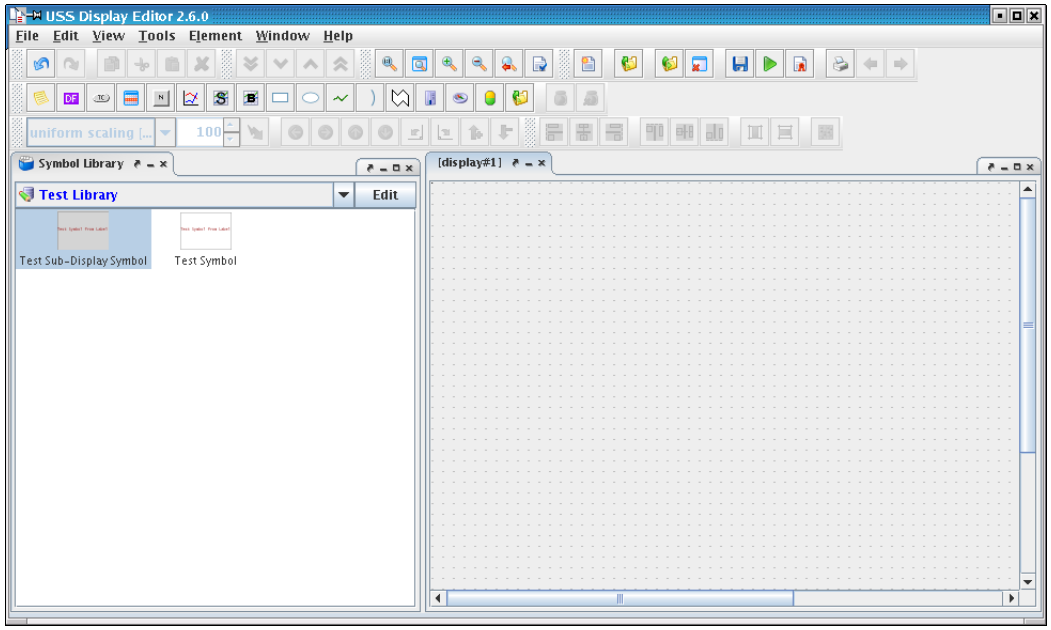
Symbol is created as a Sub Display. Sub Display are real USS-displays that can be embedded in the symbol, this powerful feature, allows for high versatility in creating symbol.

20. Delete created elements in the display, by selecting from menu: Edit | Delete



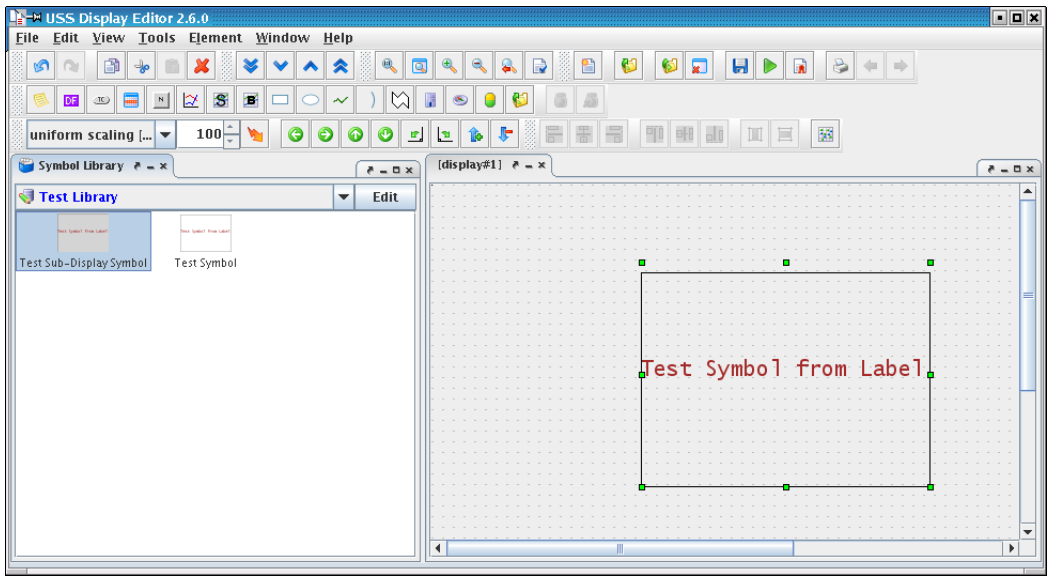
Label and rectangle are deleted.

21. Select the created symbol: Test Sub-Display Symbol in the symbol Library.



Symbol selected.

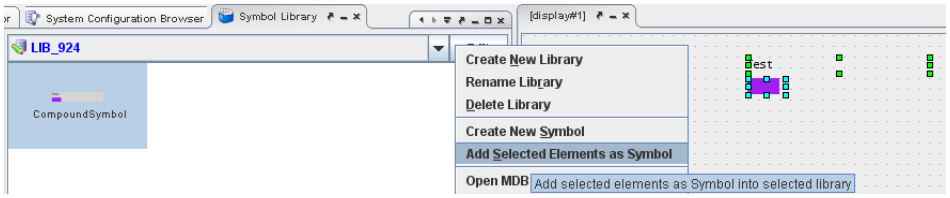
22. Left-click (hold) on selected symbol and drag it to the middle of the open display to add it.



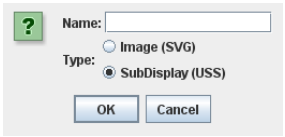
The symbol containing a sub-display is added; inside the sub-display are the label and rectangle elements.

TIP

When creating a symbol from selected elements, this can be done in two distinct ways: As a svg picture (classical way), and as a USS sub-display, were the components are kept and can be recreated.



Popup menu in symbol library view



Dialog for creating symbols from elements

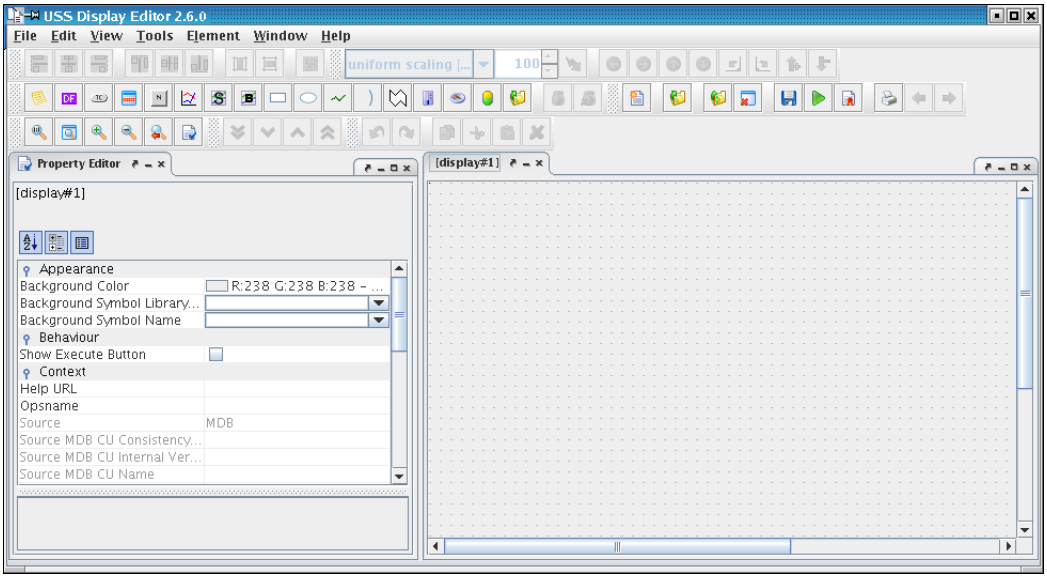
3.15 Use advanced elements

The USS package feature several advanced display elements, which give powerful visualisation possibilities.

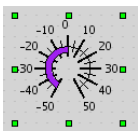
3.15.1 Lesson in creating advanced elements

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

2. Open the Property Editor from menu: View | Property Editor



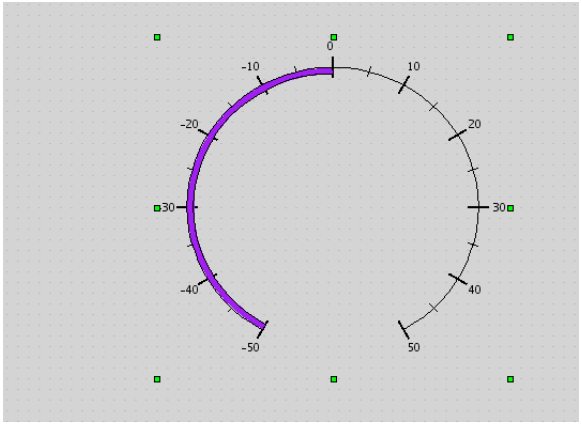
3. Make a new Elliptic Meter by selecting from menu: Element | Add | Elliptic Tick Meter



The elliptic formed meter is created in display

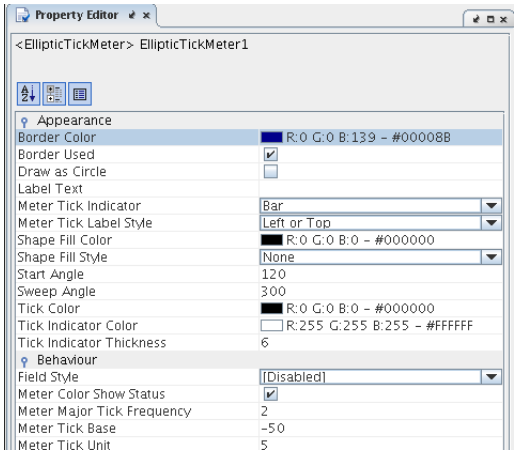
4. Undock property editor and move it, so the display can be fully seen (without overlapping from property editor).

5. Enlarge meter by dragging its corner-control-points and move it to the middle of the display by left-clicking (hold) on the Elliptic Tick Meter and dragging it.

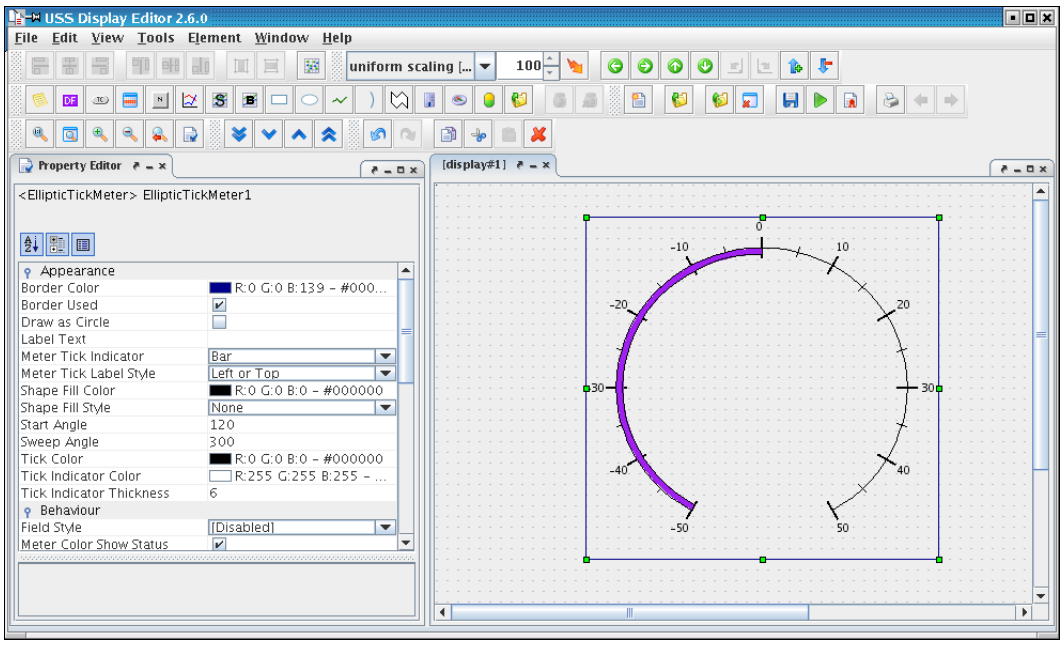


Elliptic Tick Meter is resized and located approximately in the middle of display

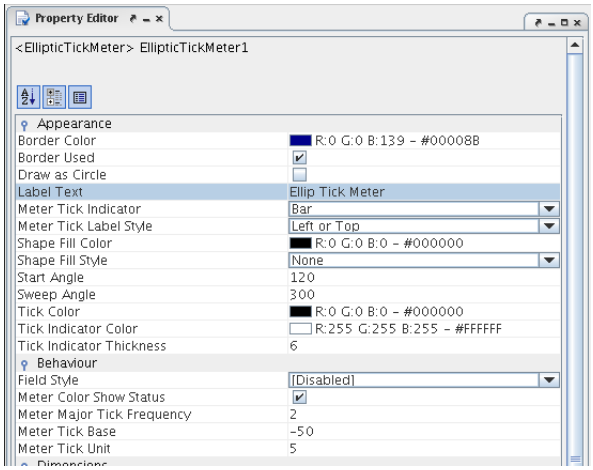
6. In the property Editor set the property: Border Used, to checked and set border color to R: 0, G: 0 and B: 139 (a blue colour).


Property Editor
<EllipticTickMeter> EllipticTickMeter1
Appearance
Border Color R:0 G:0 B:139 - #00008B
Border Used
Draw as Circle
Label Text
Meter Tick Indicator Bar
Meter Tick Label Style Left or Top
Shape Fill Color R:0 G:0 B:0 - #000000
Shape Fill Style None
Start Angle 120
Sweep Angle 300
Tick Color R:0 G:0 B:0 - #000000
Tick Indicator Color R:255 G:255 B:255 - #FFFFFF
Tick Indicator Thickness 6
Behaviour
Field Style [Disabled]
Meter Color Show Status
Meter Major Tick Frequency 2
Meter Tick Base -50
Meter Tick Unit 5

7. Re-dock the property editor.

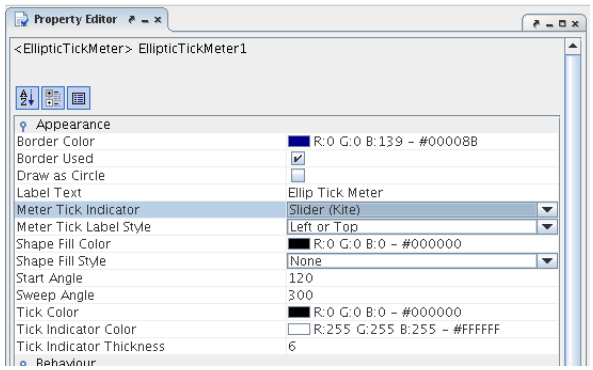


8. In property editor set property: Label Text, to Ellip. Tick Meter



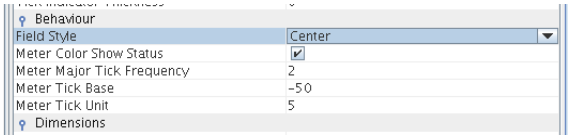
Label is shown in meter

9. In property editor set property: Meter Tick Indicator, to Slider (Kite)



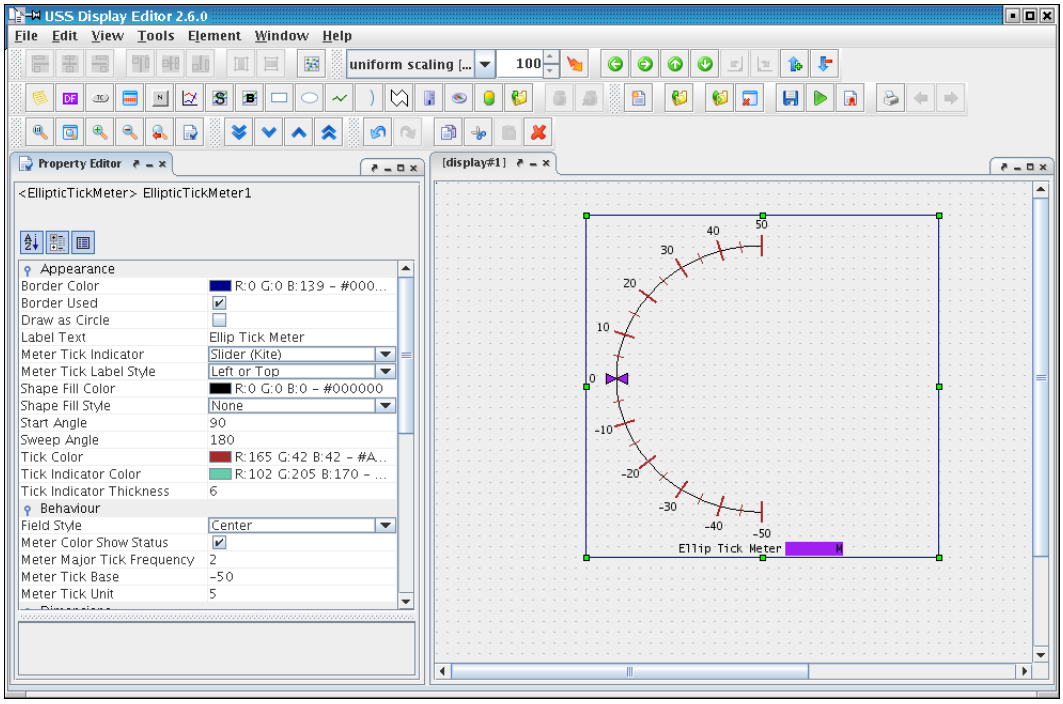
Indicator is changed

10. In property editor set property: Field Style, to Center



Field is now shown and centered

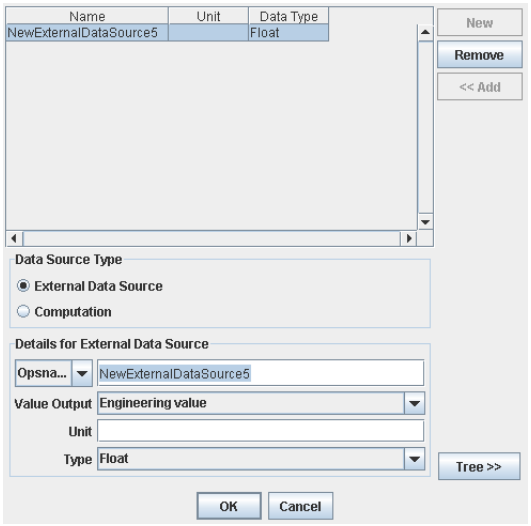
11. In property editor set the following properties: Start angle: 90 Sweep angle: 180 Tick Color: R: 165, G: 42, B: 42 Tick Indicator Color: R: 102, G: 205, B: 170



The Elliptic Tick Meter changes appearance and meter-range is changed

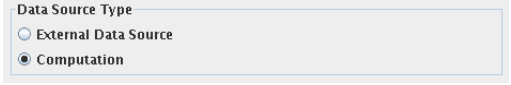
12. Select data source field in the property editor.

13. Click the property button: '...', to open the data source configuration dialog.



The data source dialog is used by the editor to edit all data sources related to display elements

14. In the data source dialog, change the Data Source Type to Computation

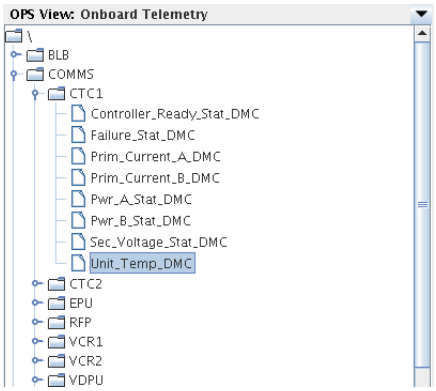


Details are changed to computation details

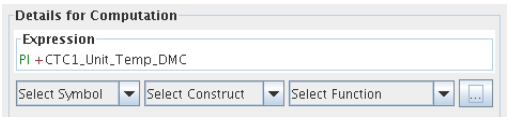
15. Select from drop-down-box (ComboBox) with text: Select Symbol, the expression-symbol: PI.

16. Click in Expression field and type on keyboard: + (plus-key)

17. Open the System Configuration Browser by clicking the button: Tree » Select the path-location: \Comms\CTC1\Unit_Temp_DMC (OPS View: Onboard Telemetry)

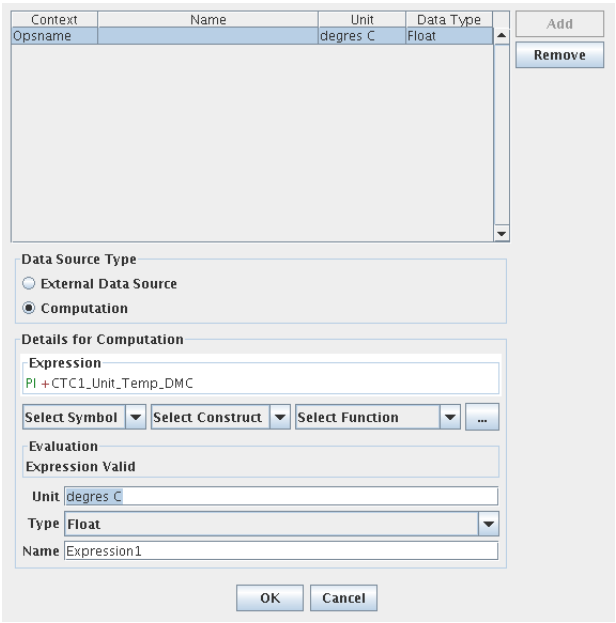


18. Click button: Add, followed by button: Close.



Label and rectangle are selected

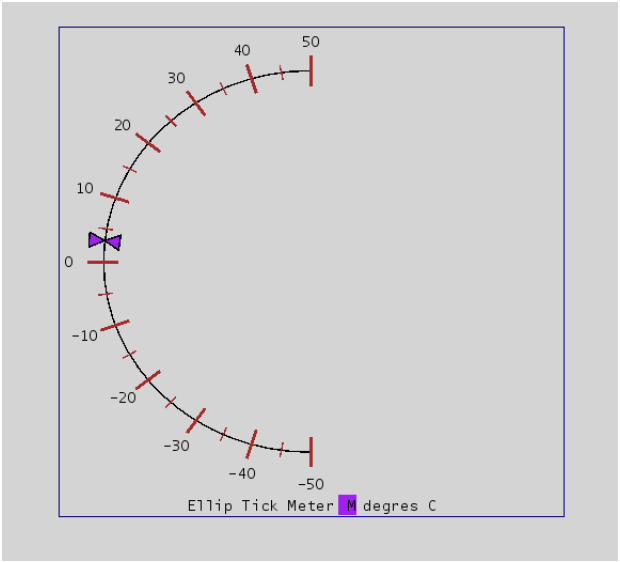
19. In the data source dialog set the Unit to: deg C



Unit is changed, the unit is a textual property of the data source

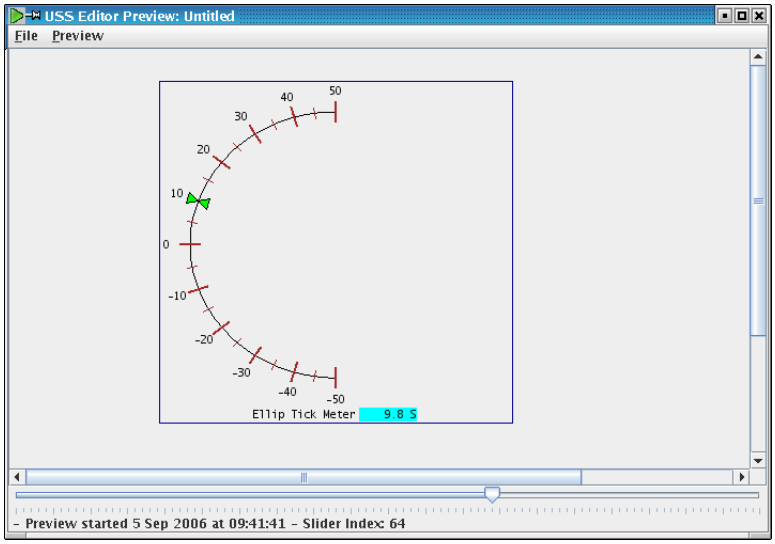
20. Click button: OK to change data source.

21. Open previewer from menu: Tools | Previewer



The created Elliptic Tick Meter is previewed

22. Use the slider at the bottom of the open dialog, to simulate values given to the data source and the effect on the meter..

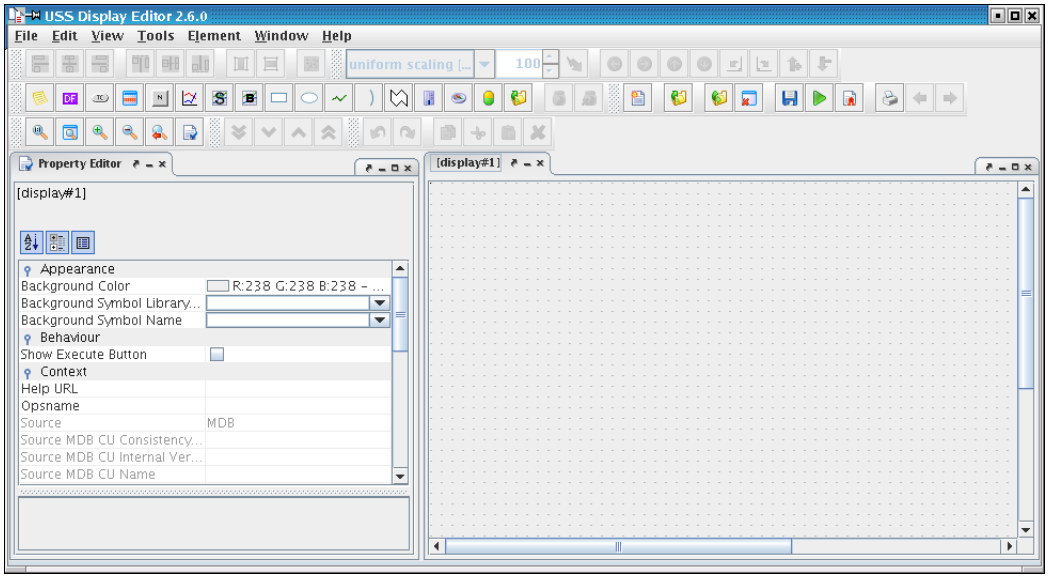


The indicator moves as the slider is moved

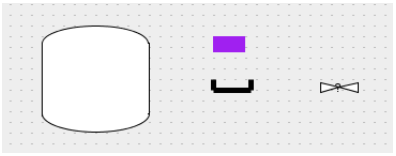
3.15.2 Lesson in creating using fluid elements

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

2. Open the Property Editor from menu: View | Property Editor



3. Make a new TankMeter, Pipe, Valve and Data field arrange approximately as picture below

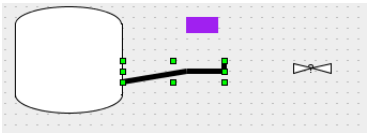


Elements for lesson: TankMeter, Pipe, Valve and Data field

4. Double-click with the mouse on the pipe, to select the left-most pipe-segment, to edit it

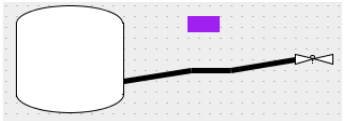
Selected pipe-segment

5. Use the mouse to move the left-top point to the TankMeter



Selected pipe-segment moved to approximately conenct with TankMeter

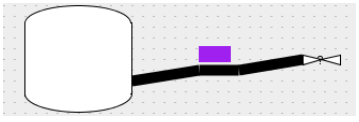
6. Do the same with the right-most pipe-segment and connect the pipe to the valve (valve maybe need arranging also)



Selected pipe-segment moved to approximately conenct with Valve

7. In property editor set pipe property: Diameter, to 10

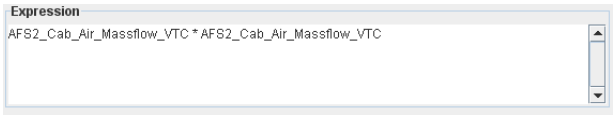
8. Arrange Data field and Valve to get approximately the layout below



Layout of display elements

9. In the property editor open the data source dialog for the TankMeter (click button: ... for property: Data Source)

10. In the data source dialog change the data source to computation and enter expression: AFS2_Cab_Air_Massflow_VTC * AFS2_Cab_Air_Massflow_VTC (Path: \APM\FLTSYS\ECLSS\AFS2\CAB_AIR_MASSFLOW)

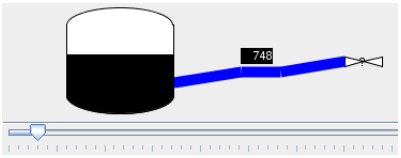


TankMeter Expression

11. In the property editor open the data source dialog for the Data Field (click button: ... for property: Data Source)
12. In the data source dialog change the data source to computation and enter expression (derivative of the TankMeter expression): 2 * AFS2_Cab_Air_Massflow_VTC
13. For the TankMeter, Pipe and Valve change the property: Fluid, to 'Air'

Fluid Property changed to Air

14. Click and select only the pipe
15. Click the greyed icon for the property fluid, to open the data source configuration dialog, and add new data source, and enter the expression: if 2 * AFS2_Cab_Air_Massflow_VTC > 900 then "Red" else "Blue"
16. Click and select only the TankMeter
17. Change Maximum Range of TankMeter to 250000
18. Open the previewer via menu: Tools | Preview Display...
19. Use slider to simulate display



Preview of created display

3.16 Change USS Properties File

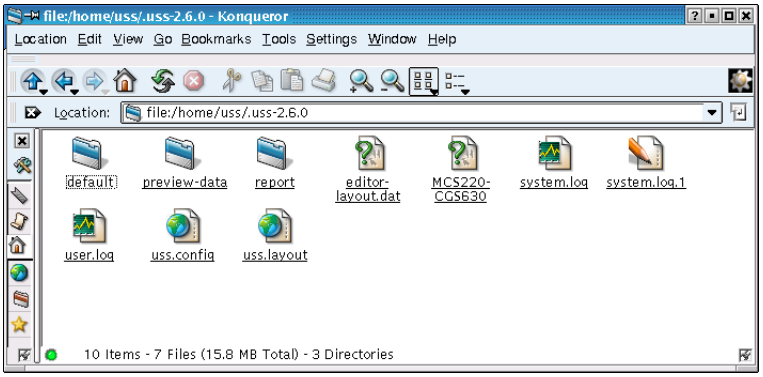
The USS package feature a common configuration system, which is located in the home folder in the folder: .uss-<USS_VERSION> (EX: .uss-2.6.0)

3.16.1 Lesson in USS properties

The USS Editor and Executor use Data Quality Indicator definition to give information about the state of the data being processed.

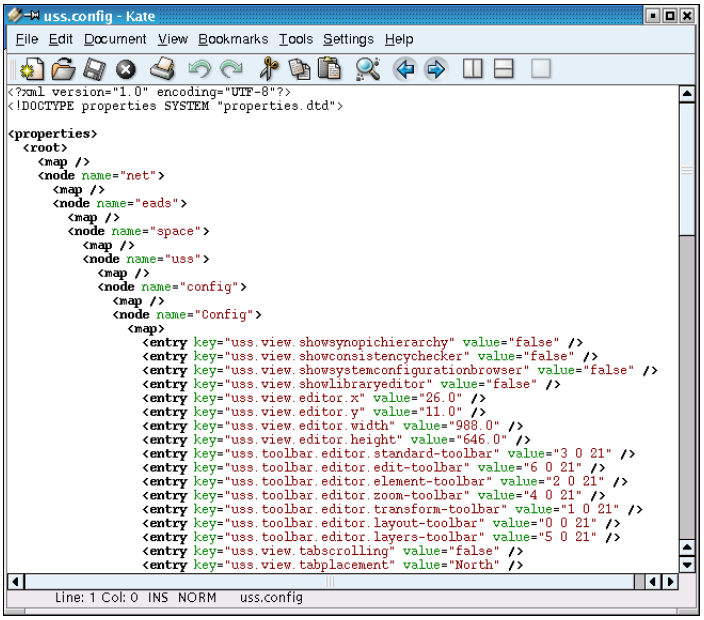
the lesson assumes the XML editor KXML-editor is installed, as well as the Kate (KDE editor).

1. Opening a file browser to the >HOME_DIRECTORY</.uss-<USS_VERSION> (EX: .uss-2.6.0) directory. Here open the file: uss.config (you might need to change file-filtering to All-Files)



Folder contains many files, i.e. log-file, the folder also contains the uss.config file, which contains the settings for USS.

2. Opening the Kate-editor and browser to the >HOME_DIRECTORY</.uss-<USS_VERSION> directory. Here open the file: uss.config.



The Kate-Editor shows the content of the XML-formatted uss.config file for the USS system

The Kate editor contains features for context highlighting. Browse down to see the USS properties, they can be changed manually by editing via Kate.

3. Close the Kate editor again.

4. Opening the KXML-editor and browser to the >HOME_DIRECTORY</.uss-<USS_VERSION> directory. Here open the file: uss.config.

The KXML-Editor shows the content of the XML-formatted uss.config file for the USS system

The XML tree hierarchy is shown on the left and the content of the selected node on the right, where the content can be edited.

5. Browse in the XML tree hierarchy to the node: properties/root/node/node/node/node/node/node/map/entry(1) and select it to show the content.

	Namespace	Name	Value
1		key	uss.view.editor.x
2		value	0.0

Changing the values here will change view settings X value of the editor, which can be edited through the Editor Preferences also.

6. Try changing the value to 25.0, to set the Editors default view setting for the X value.

	Namespace	Name	Value
1		key	uss.view.editor.x
2		value	25.0

The value is changed, and will be used in future invocations of the editor

The value is changed, and will be used in future invocations of the editor. If editor is open it has to be restarted to load the new value.

TIP

Generally most properties should be controlled via the editor or executor (uss.config)

TIP

A modified properties file can be used to ease distribution of the USS tools

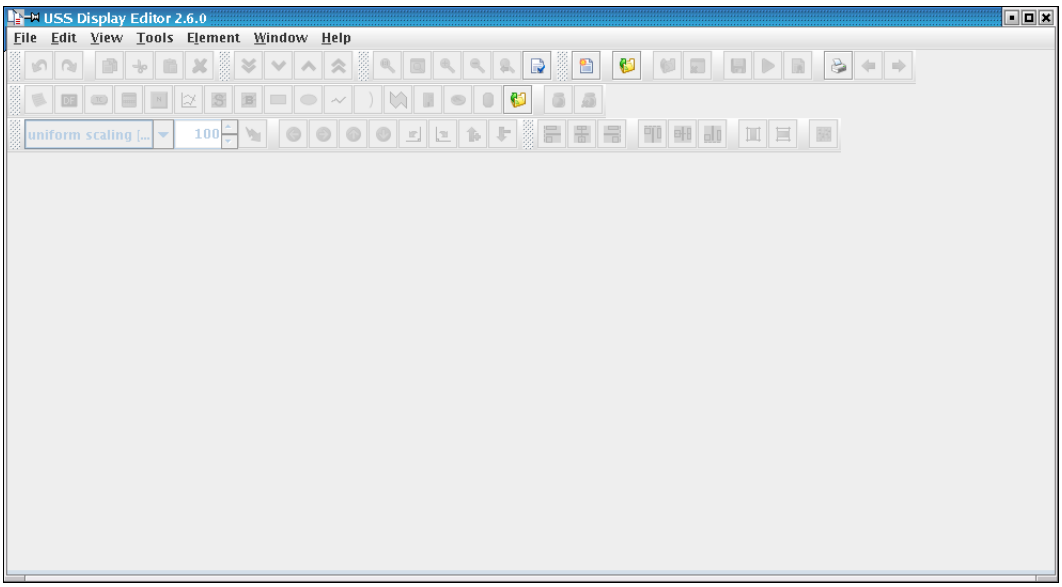
3.17 Select SCOE Files

The USS package uses a SCOE file, to ease creation of end-items.

3.17.1 Lesson in changing the SCOE file

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

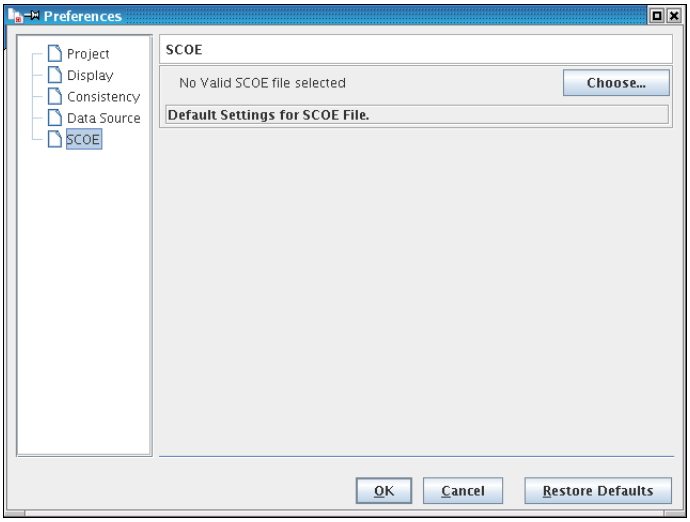
2. Close the open default new display, to allow changing of the SCOE file.



No display open

3. Open the editor preferences by selecting: Edit > Preferences.

4. Left-click on SCOE.



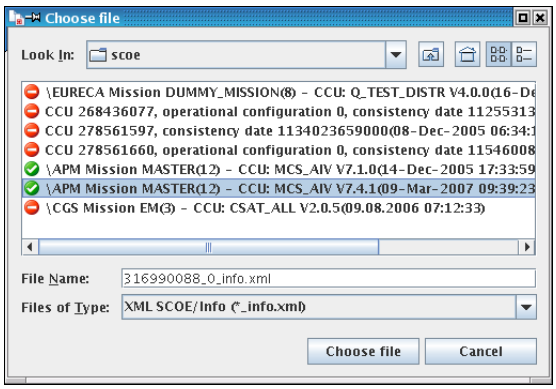
5. Click button: Choose..., which opens the SCOE file chooser. The SCOE file chooser displays the scope details for which the SCOE file set was generated for.

6. Select a new SCOE file. The icon in front of the SCOE file set indicates whether it is compatible with the currently used systeminterface version or not. In case of incompatibility:

   - the loading of the SCOE file set may fail,
   - the loading may succeed but problems may occur later,
   - or possibly no problems may arise.

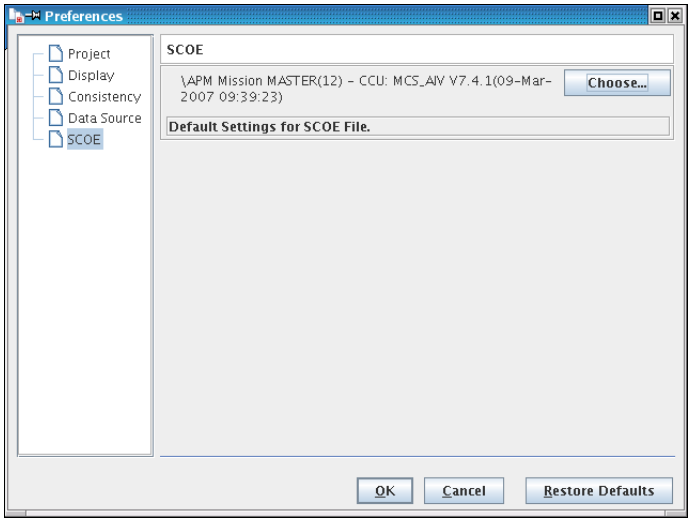
IMPORTANT

It is not recommended to load SCOE file sets which have been marked as incompatible. Use is at your own risk.


Choose file
Look In:
scoe
\EURECA Mission DUMMY_MISSION(8) - CCU: Q_TEST_DISTR V4.0.0(16-De
CCU 268436077, operational configuration 0, consistency date 11255313
CCU 278561597, consistency date 1134023659000(08-Dec-2005 06:34:1
CCU 278561660, operational configuration 0, consistency date 11546008
\APM Mission MASTER(12) - CCU: MCS_AIV V7.1.0(14-Dec-2005 17:33:59
\APM Mission MASTER(12) - CCU: MCS_AIV V7.4.1(09-Mar-2007 09:39:23
\CGS Mission EM(3) - CCU: CSAT_ALL V2.0.5(09.08.2006 07:12:33)
File Name:
316990088_0_info.xml
Files of Type:
XML SCOE/Info (*_info.xml)
Choose file
Cancel

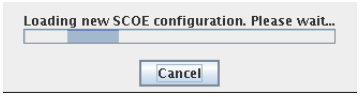
New SCOE file selected

7. Click button: Choose file.



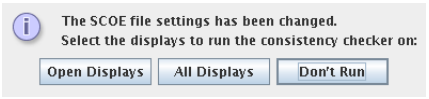
The new SCOE file is now choosen

8. Click button: OK, to change the editor preferences (uss.properties file). The editor will reload the System Configuration Browser (based on the SCOE file)



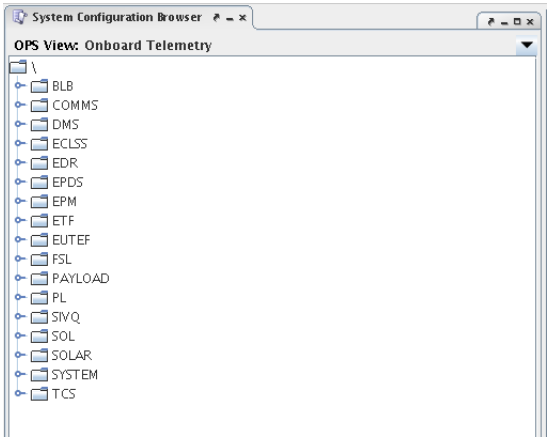
Reloading of new SCOE file takes some seconds

9. A confirmation box pops-up where the consistency checker can be rerun on open displays or all displays (which are those displays who are under the project folder). Also it is possible to choose not to run the consistency checker.


The SCOE file settings has been changed.
Select the displays to run the consistency checker on:
Open Displays
All Displays
Don't Run

10. Click Don't Run.

11. SCOE file is reloaded in the System Configuration Browser.



3.18 Check Consistency

The USS Editor can check the consistency of created displays against the configured SCOE file and there by save time for display developers.

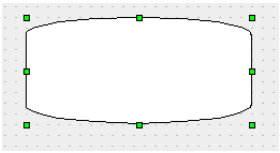
3.18.1 Lesson in checking display consistency

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

2. Add a label (menu: Element | Add | Label) Double-click (left-mouse-button) to edit text in line and write text: Label for check

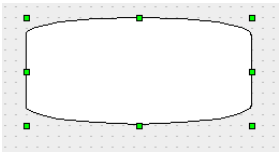
Label created and text changed

3. Add a Tank Meter (menu: Element | Add | TankMeter)



Tank Meter created

4. Open Property Editor (Menu: View | Property Editor)

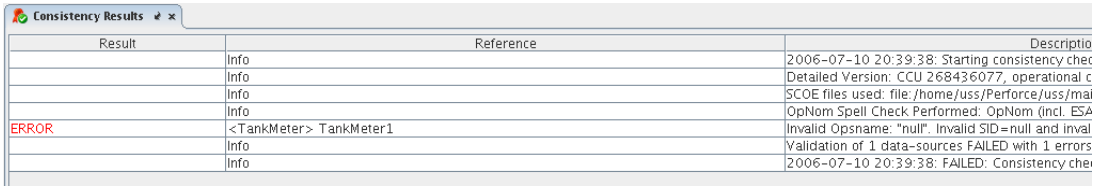


The properties for the advanced element Tank Meter is shown

5. Choose from menu: Tools | Check Consistency.

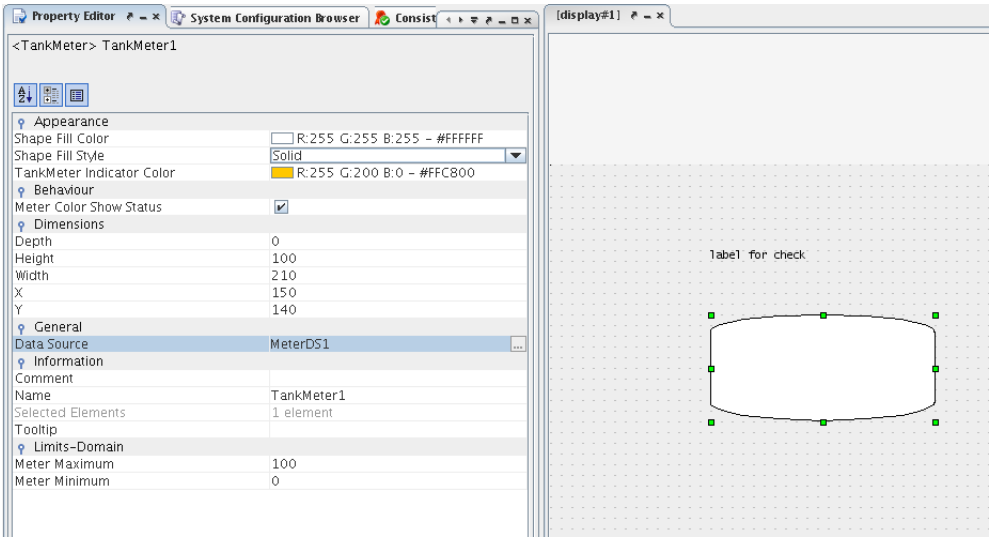
A dialog opens stating display is inconsistent and there is one problem

6. Click button: OK and the tool-view: Consistency Result opens.



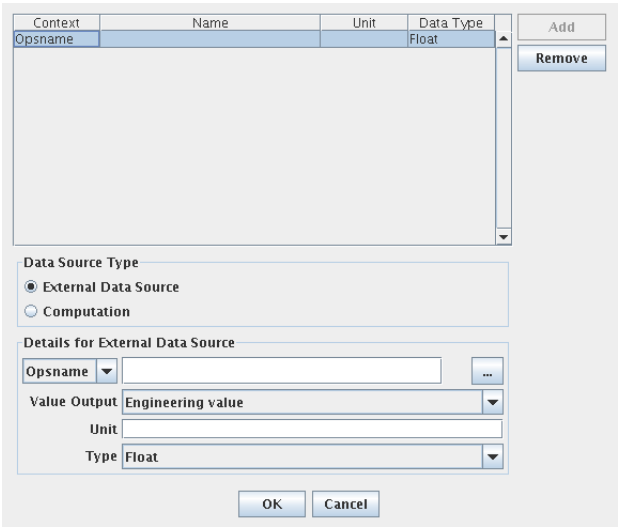
The consistency results shown the information processed by the consistency checker and the results. One error is marked red, stating that the Tank Meter has a invalid data source with invalid parameters: OpsName, SID and pathname

7. Select/focus the Property Editor again, and select the property: Data Source.



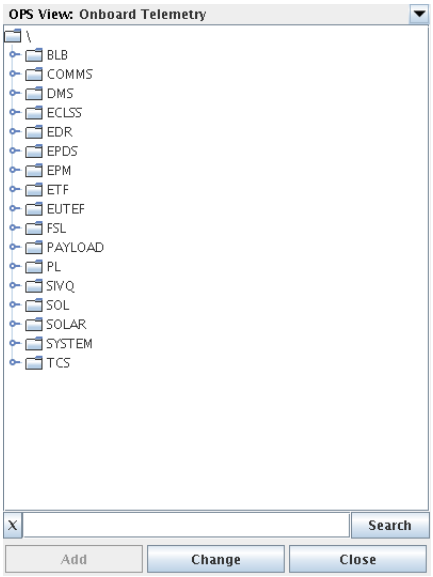
Property: Data Source highlighted

8. Click button: ..., to change open the data source dialog.



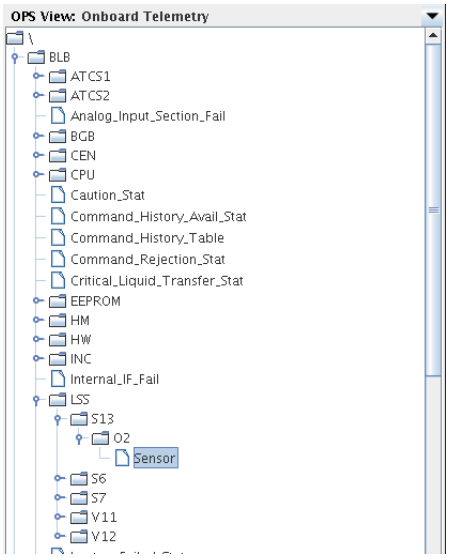
Data source dialog opens showing the data source attached to the Tank Meter

9. Click under Details for External Data Source on button: *'...'*, to open the System Configuration Browser.



System Configuration Browser opens (in OPS View: Onboard Telemetry)

10. In the search field (left of button: Search) type: 'Sensor', and click button: Search.

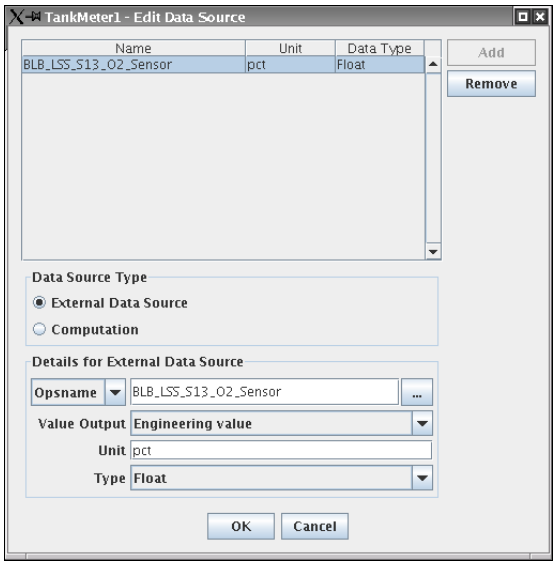


System Configuration Browser searches and selects the end-item: \BLB\LSS\S13\O2\Sensor

11. Click button: Change.

System Configuration Browser changes the data source in the Data Source Dialog

12. Click button: Close.



Data Source Dialog now has a fully configured data source

13. Click button: OK, in data source dialog.

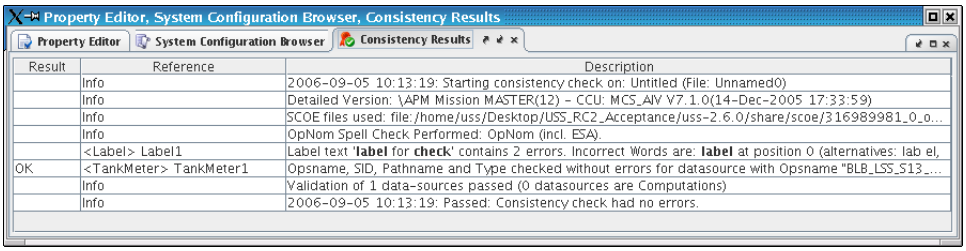
Shown Data Source in property editor is now changed

14. Now check consistency again (Tools | Check Consistency).

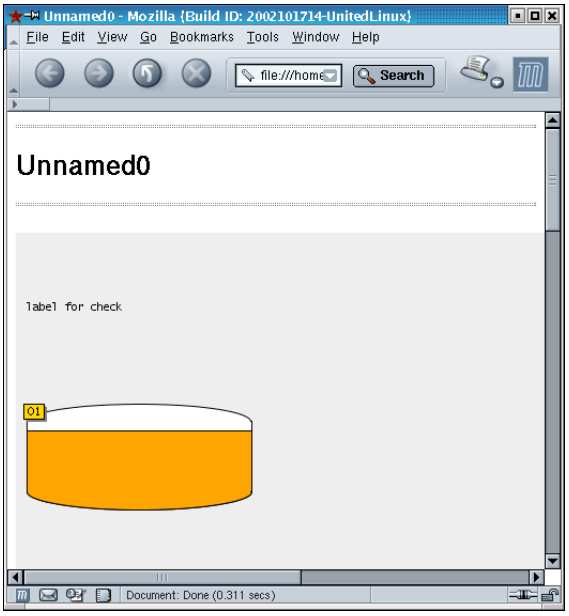
Dialog shows that display is now consistent

15. Click button: 'OK', to close confirmation dialog.

16. Select/focus the Consistency Results View again.


Property Editor, System Configuration Browser, Consistency Results
Property Editor
System Configuration Browser
Consistency Results
Result
Reference
Description
Info
2006-09-05 10:13:19: Starting consistency check on: Untitled (File: Unnamed0)
Info
Detailed Version: \APM Mission MASTER(12) - CCU: MCS_AIV V7.1.0(14-Dec-2005 17:33:59)
Info
SCOE files used: file:/home/uss/Desktop/USS_RC2_Acceptance/uss-2.6.0/share/scoe/316989981_0_0...
Info
OpNom Spell Check Performed: OpNom (incl. ESA).
<Label> Label1
Label text 'label for check' contains 2 errors. Incorrect Words are: label at position 0 (alternatives: lab el,
OK
<TankMeter> TankMeter1
Opsname, SID, Pathname and Type checked without errors for datasource with Opsname "BLB_LSS_S13_...
Info
Validation of 1 data-sources passed (0 datasources are Computations)
Info
2006-09-05 10:13:19: Passed: Consistency check had no errors.

17. To generate a HTML report, select from menu: File | Generate HTML report



TIP

The Editor property: OpNom check can be enabled to check via the consistency checker, the conformance of label-elements attached to f.ex. data fields. The text in this type of attached label has to comply with the OPS name of the data source involved (disable the check in editor preferences and non-OpNom labels will not be reported in the consistency-check).

3.19 MDB Displays

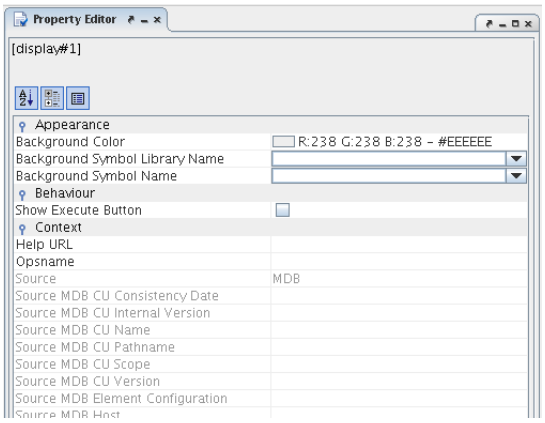
The USS Editor can work with a MDB. Displays can be synchronized (normally and forced/overwrite), submitted (new displays / check-in of changes) and Deleted.

Lesson setup: A working connection to a MDB is needed, proper setup of editor preferences to the MDB in question.

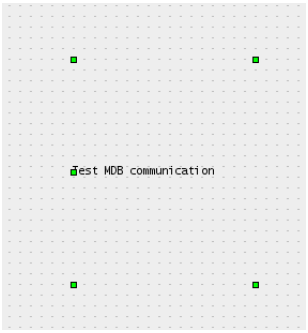
3.19.1 Lesson in using the editor MDB interfacing

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout

2. Open Property Editor from menu: View | Property Editor

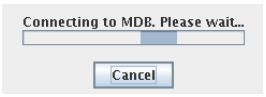


3. Make a label (Menu: Element | Add | Label) and set property: Label Text, in property editor to: Test MDB Connection.



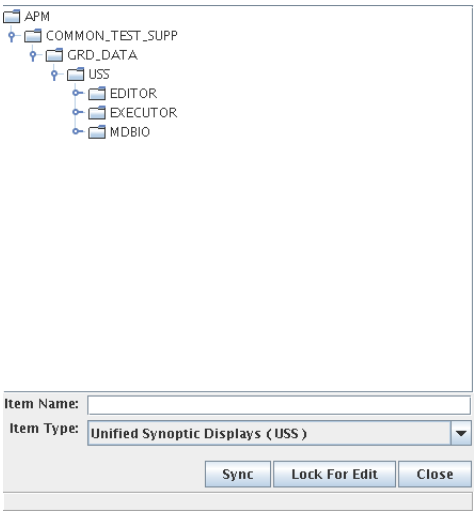
Label changed

4. Select from menu: File | MDB | MDB Browser

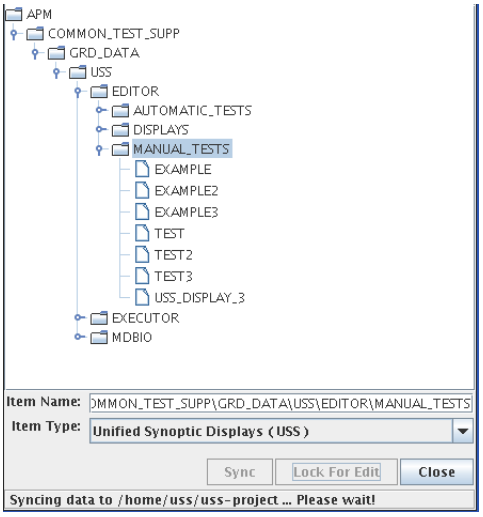


The MDB Browser is loading configuration data from the MDB (can take minutes)

5. After loading of MDB configuration data, the MDB browser dialog opens.

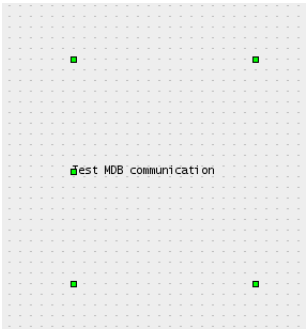


6. Open and select the MDB path: '\APM\COMMON_TEST_SUPP\GRD_DATA\USS\EDITOR\MANUAL_TESTS', and press button: Sync.



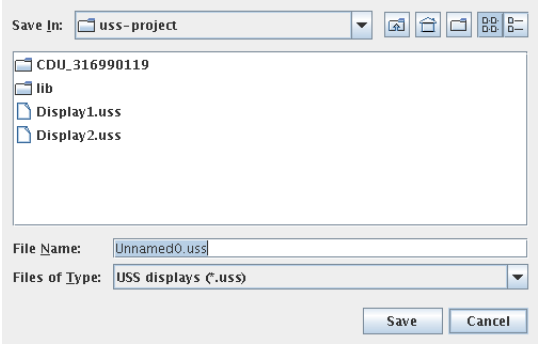
The MDB Browser is synchronizing the MDB node: '\APM\COMMON_TEST_SUPP\GRD_DATA\USS\EDITOR\MANUAL_TESTS', as well as all the sub-nodes and displays

7. Close MDB Browser (press button: 'Close').



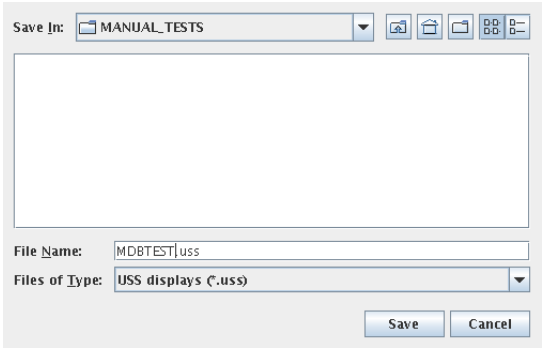
Dialog closes

8. Save display by selecting from menu: 'File|Save'.



The save dialog appears, opened in the uss-project folder

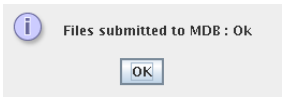
9. Choose save-path: '<uss-project-directory>/<CU_DIRECTORY>/APM/COMMON_TEST_SUPP/GRD_DATA
And file-name: 'MDBTEST.uss'



10. Click button: Save, to save file.

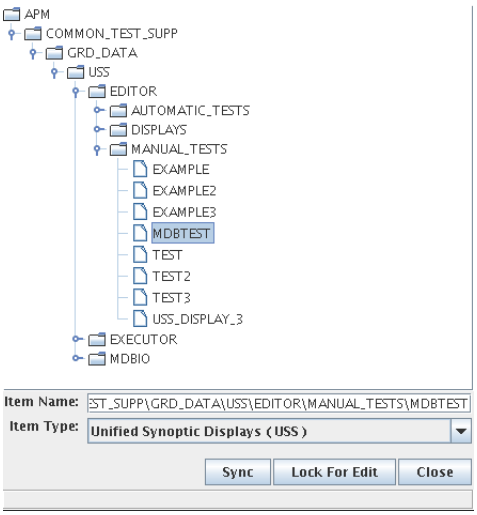
Display is saved in correct path MDB synchronization

11. To add display to MDB, choose from menu: File | MDB | ADD to MDB.



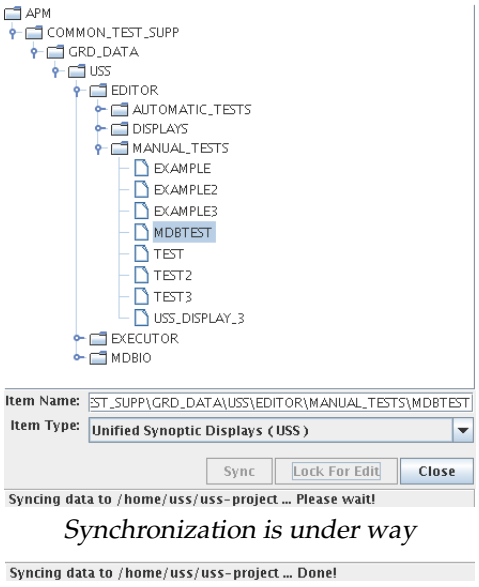
Confirmation is shown, when display is submitted

12. To verify the MDB save of the new display, choose from menu: File | MDB | MDB Browser And browse to the MDB-path: \APM\COMMON_TEST_SUPP\GRD_DATA\USS\EDITOR\MANUAL_TESTS



MDB Browser shows saved display

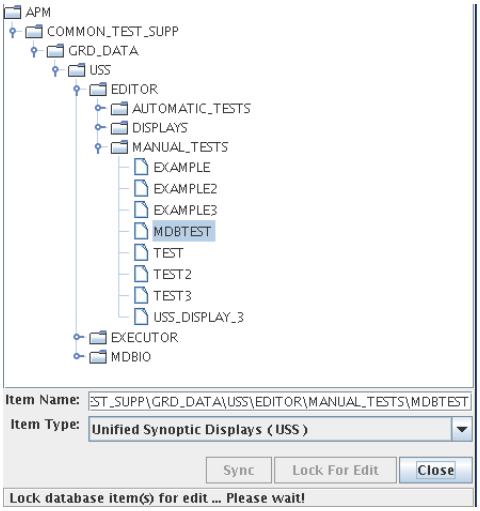
13. Select the MDB display: MDBTEST, and press button: sync.



Synchronization is under way

Synchronization is done

14. Now open the display again for editing by clicking the button: 'Lock For Edit', in the MDB Browser.

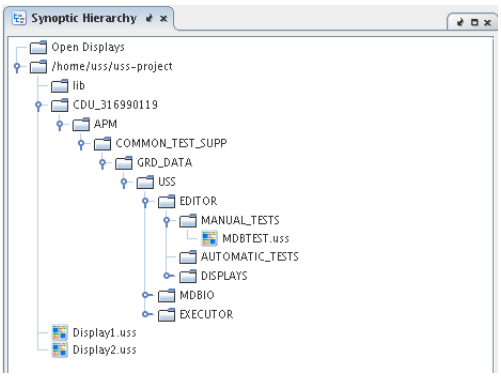


MDB Browser is locking the display

MDB Browser is done locking the display

15. Click button: Close, in MDB Browser.

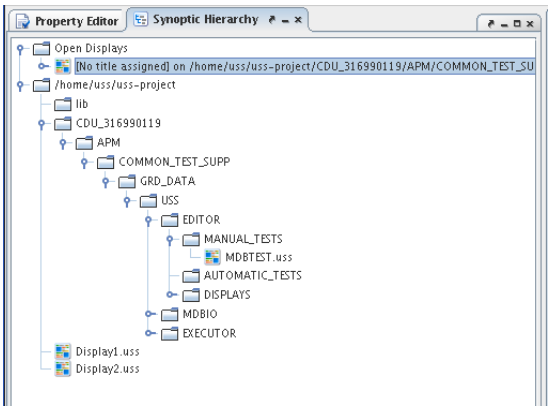
16. Close the open display, selecting from menu: File | Close. Open Synoptic Hierarchy by selecting from menu: View | Synoptic Hierarchy.



Synoptic Hierarchy shows the project-files

17. Browse to the saved display in the project-folder via the Synoptic Hierarchy, path: '<uss-project-directory>/<CU_DIRECTORY>/APM/COMMON_TEST_SUPP/GRD_DATA/USS/EDITOR/MANUAL_TESTS/M

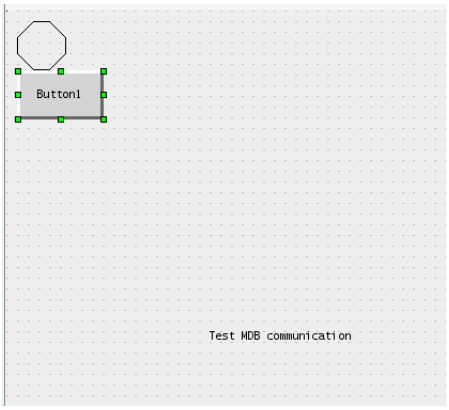
18. Right-click on project-display: MDBTEST.uss, and choose from pop-up menu Open display.



Display is reopened

19. Add a polygon from menu: Element | Add | Polygon.

20. Add a navigation button from menu: Element | Add | Navigation Button.

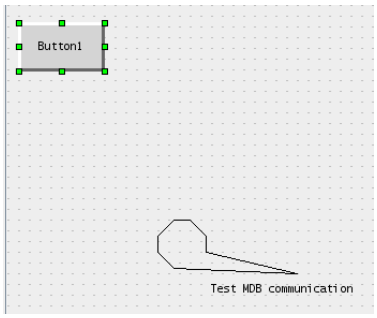


The display has additional elements: polygon and navigation button

21. Select the polygon by left-clicking on it.

22. Move the polygon by left-click (hold) and dragging it.

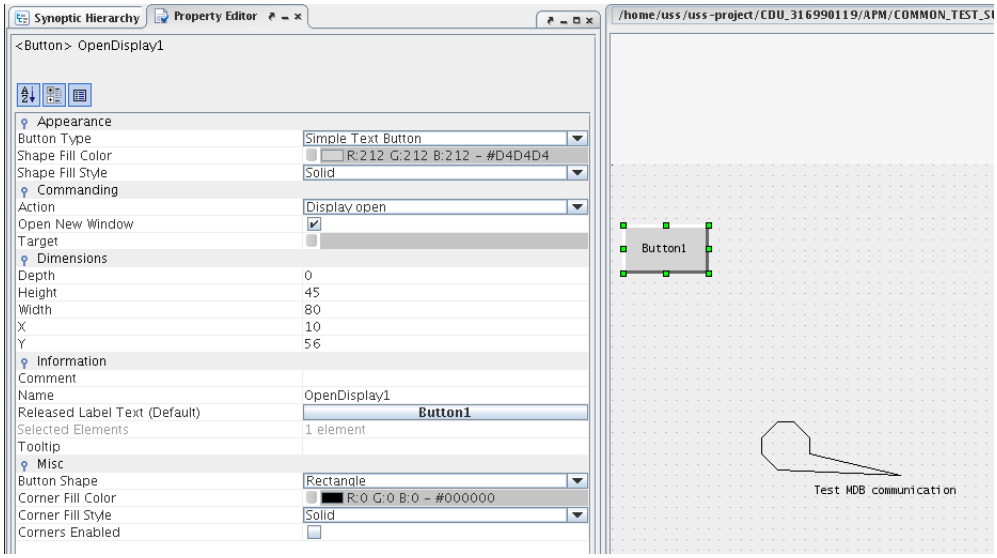
23. Left-click (hold) and move one of the control-points of the polygon, to change its shape.



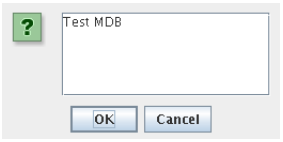
Polygon is changed

24. Open the Property Editor by selection menu: View | Property Editor.

25. Select the button by left-clicking on it.



Button properties are shown in the Property Editor

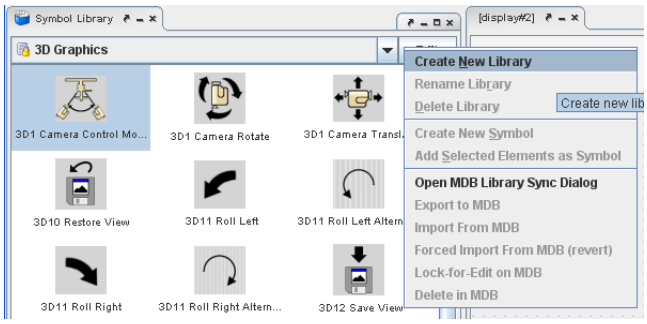
26. Click on the button: Button1, for the property: Released Label Text to open text edit dialog.
27. 

Dialog for multi-line text edit shows new text

28. Click button: OK.
29. Save display by choosing from menu: File | Save.
30. Submit the changes to the MDB by selecting from menu: File | MDB | Save in MDB.
31. Delete the display file in the MDB, by selecting from menu: File | MDB | Delete in MDB.
32. Click button: Yes, to delete the tutorial display-file.

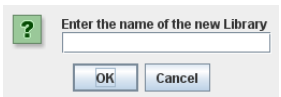
3.19.2 Lesson in using the editor MDB interfacing for Symbol Libraries

1. Start by opening the editor via installed icon, close all tool-views. Screenshot of USS Editor started in default layout
2. Select the view: Symbol Library (if not open, use menu: View | Symbol Library)
3. In the Symbol Library use the button: Edit, to create a new user library



Popup menu for Symbol Libraries

4. Select New Library

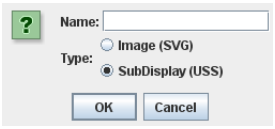


Dialog to specify new library

5. Enter new library name: TEST_LIB, and press button: Ok
6. In the directory chooser select a filesystem location compliant with the MDB structure. This lesson uses: <USS-PROJECT-ROOT>/lib/CDU_316990235/APM/COMMON_TEST_SUPP/GRD_DATA/USS/EDITOR/M
7. Choose directory and press button: Open

New library created and selected in Symbol Library view

8. Create a new Label in the empty display and set text to: TEST_SYMBOL. Click on button: Edit in Symbol Library View and click in popup-menu: Add Selected Elements as Symbol



Dialog for specifying symbol shown

9. Enter new symbol name: TEST_SYMBOL, select SVG as type, and press button: Ok
10. Save library in MDB by click button: Edit, in symbol library view and popup-menu: Save in MDB

Dialog for showing progress of MDB connection

Dialog showing result of save

TIP

The Symbol Library View button: Edit, can be used to Lock-for-Edit, Save, Synchronize etc. MDB Symbol Libraries

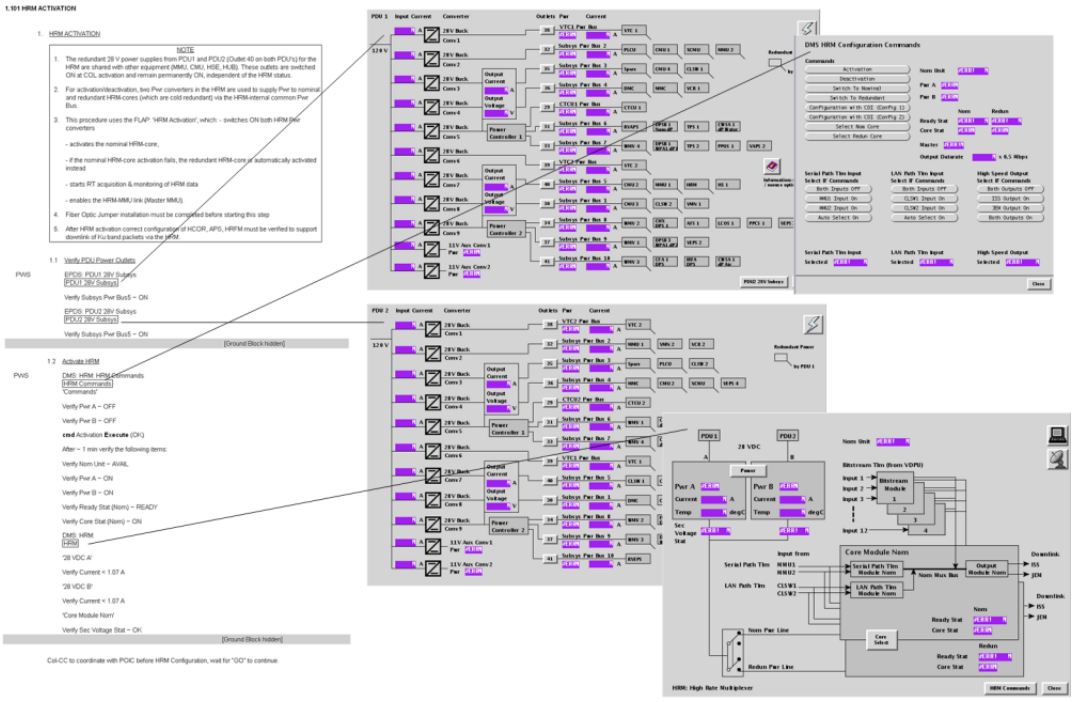
TIP

Symbol Libraries must have uppercase names of maximum 16 characters to be compliant with the MDB

3.20 Extended Example: Create A Procedural Display

The USS Editor can be used to create procedural displays in the USS display format. Procedural displays are task specific displays that visually resemble ODF procedures. They provide all the command buttons and data fields as well as all the guidance required to accomplish a given task and therefore minimizes the need for display navigation.

This example will demonstrate the creation of a simple, but realistic procedural display for activating the Columbus HRM (High Rate Multiplexer, COMMS subsystem). The ODF procedure for this task and the four conventional displays required for carrying it out are shown in the following figure:



ODF Procedure and Onboard Displays

The following figure shows the SATMON display as it will be used by ColCC:

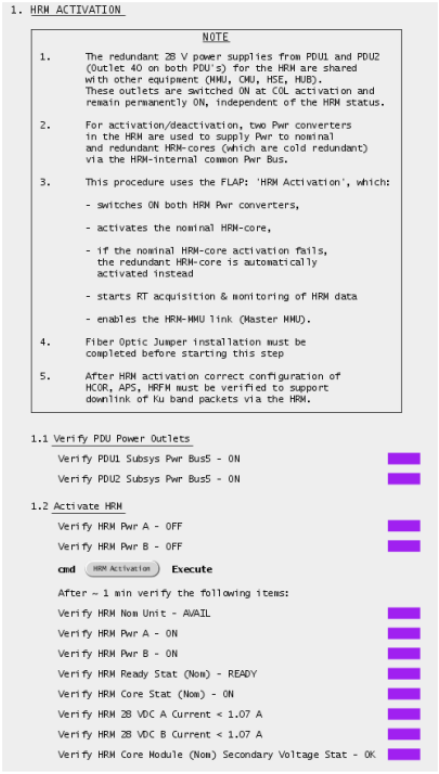
```
Step 1.:HRM Activation

Step 1.1:Verify PDU Power Outlets
Verify:
PDU1_Subsys_Pwr_Bus5_On_Off_Stat_DMC  ON
PDU2_Subsys_Pwr_Bus5_On_Off_Stat_DMC  ON

Step 1.2.:Activate  HRM
Verify:
HRM_Pwr_A_Stat_DMC                    OFF
HRM_Pwr_B_Stat_DMC                    OFF
Execute FLAP: HRM_Activation_AP
Verify:
HRM_Nom_Branch_Avail_Stat_SW          Avail
HRM_Pwr_A_Stat_DMC                    ON
HRM_Pwr_B_Stat_DMC                    ON
HRM_Nom_Ready_Stat_DMC                READY
HRM_Nom_Core_Stat_DMC                 ON
HRM_Pwr_A_Current_DMC                 < 1.07 A
HRM_Pwr_B_Current_DMC                 < 1.07 A
HRM_Nom_Sec_Voltage_Stat_DMC          OK
```

SATMON Display

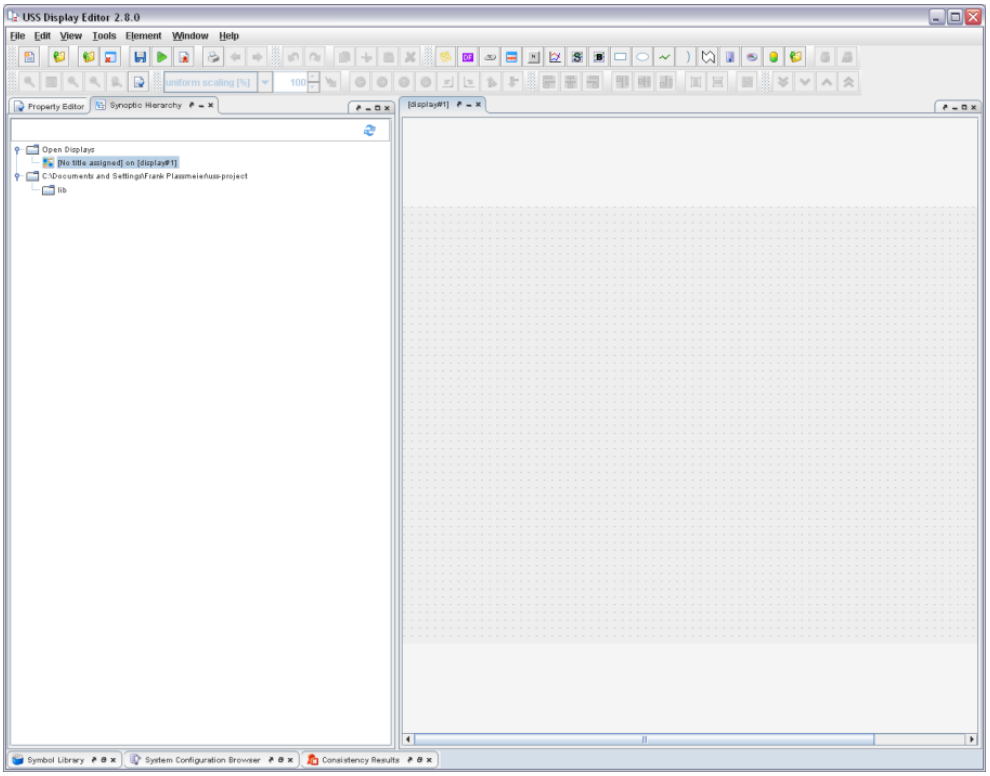
The procedural display that will be created in this lesson will contain data fields and commands that correspond to the SATMON display but look like the ones in the onboard displays. Step numbers and titles as well as procedure title and the note box at the beginning will also be present. The procedural display will look as follows:


1. HRM ACTIVATION
NOTE
1. The redundant 28 V power supplies from PDU1 and PDU2 (Outlet 40 on both PDU's) for the HRM are shared with other equipment (MMU, CMU, HSE, HUB). These outlets are switched ON at COL activation and remain permanently ON, independent of the HRM status.
2. For activation/deactivation, two Pwr converters in the HRM are used to supply Pwr to nominal and redundant HRM-cores (which are cold redundant) via the HRM-internal common Pwr Bus.
3. This procedure uses the FLAP: 'HRM Activation', which:
- switches ON both HRM Pwr converters,
- activates the nominal HRM-core,
- if the nominal HRM-core activation fails, the redundant HRM-core is automatically activated instead
- starts RT acquisition & monitoring of HRM data
- enables the HRM-MMU link (Master MMU).
4. Fiber Optic Jumper installation must be completed before starting this step
5. After HRM activation correct configuration of HCOR, APS, HRFM must be verified to support downlink of Ku band packets via the HRM.
1.1 Verify PDU Power Outlets
Verify PDU1 Subsys Pwr Bus5 - ON
Verify PDU2 Subsys Pwr Bus5 - ON
1.2 Activate HRM
Verify HRM Pwr A - OFF
Verify HRM Pwr B - OFF
cmd HRM Activation Execute
After ~ 1 min verify the following items:
Verify HRM Nom Unit - AVAIL
Verify HRM Pwr A - ON
Verify HRM Pwr B - ON
Verify HRM Ready Stat (Nom) - READY
Verify HRM Core Stat (Nom) - ON
Verify HRM 28 VDC A Current < 1.07 A
Verify HRM 28 VDC B Current < 1.07 A
Verify HRM Core Module (Nom) Secondary Voltage Stat - OK

ODF Style Procedural Display

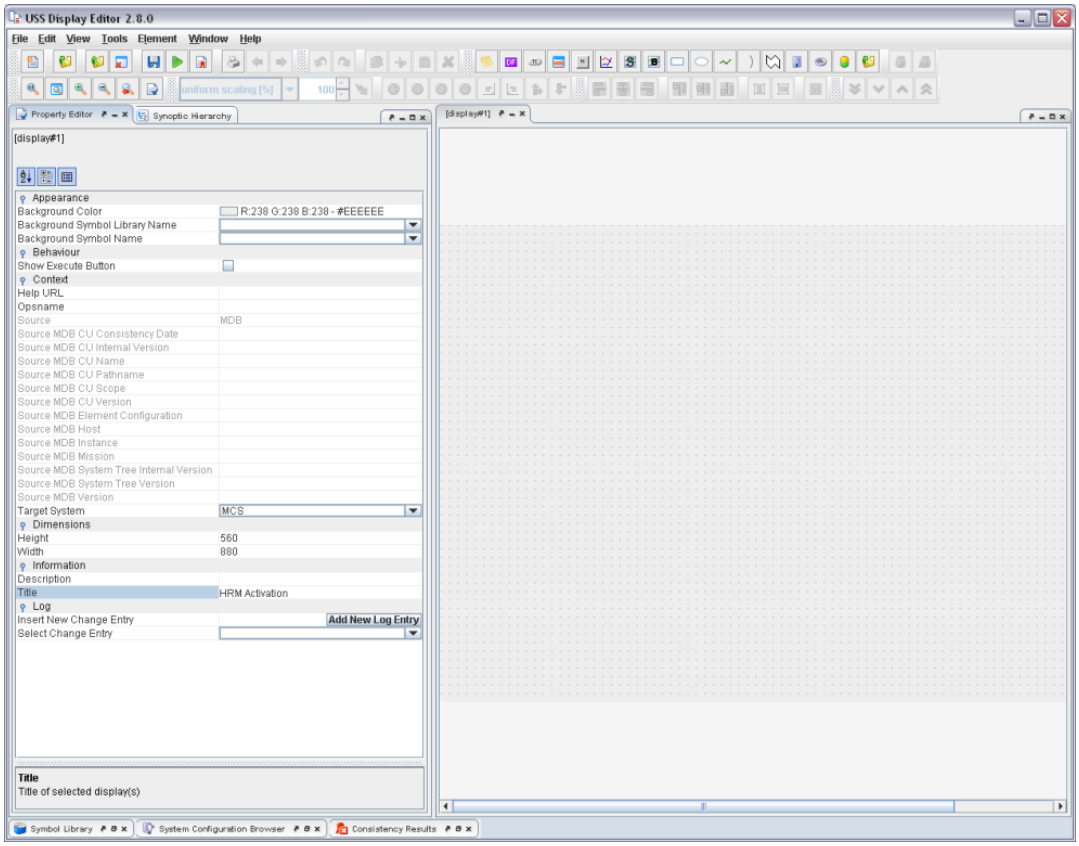
3.20.1 Lesson in creating procedural displays

1. Creating a procedural display is very similar to creating other types of displays. Start by opening the editor via installed icon.



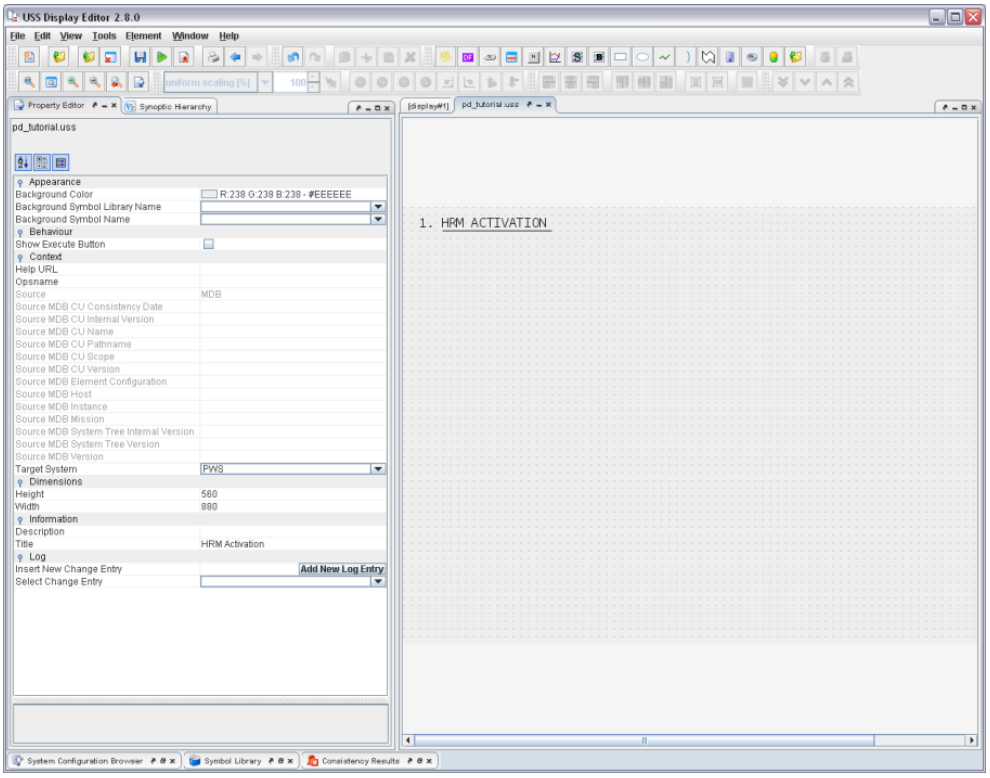
Editor started with default empty display created, two tool views open

2. Start by setting the display title. If the Property Editor tab on the left of the window is not already selected, select it. In the pane corresponding to that tab, left-click in the field named Title, and enter the display title.



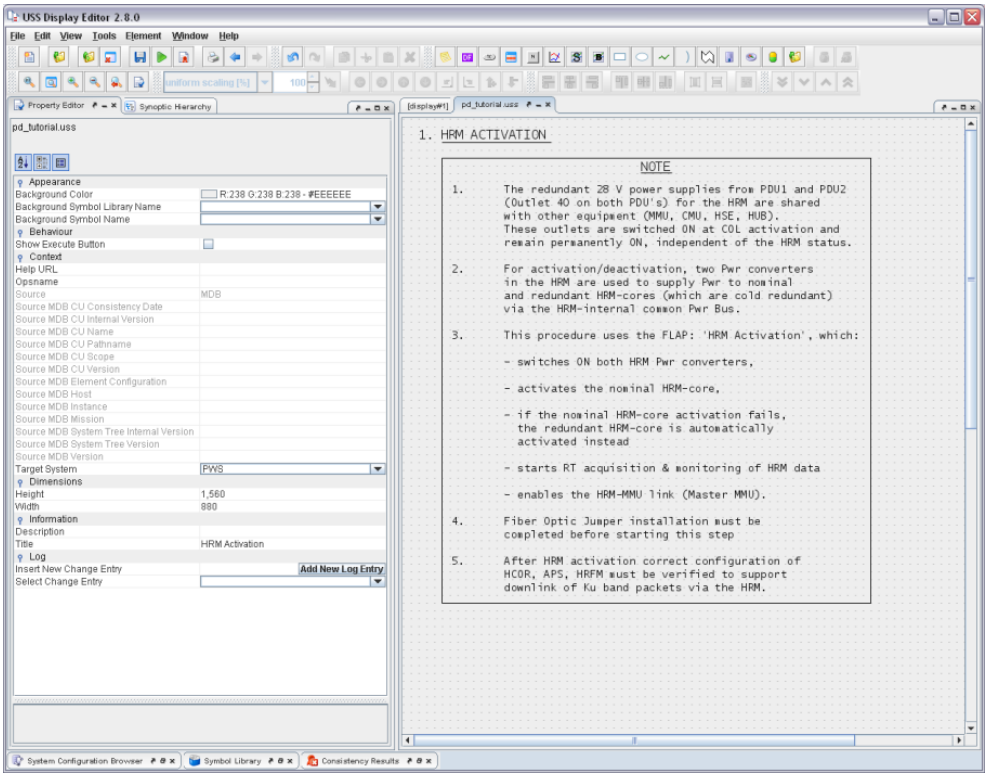
Editor after entering display title

3. Continue by adding the first (and for the simple example only) main step title as a label. Right-click on the background of the display, then select Add Element | Label from the menu that appears. A new label is created and automatically selected. If the Property Editor tab on the left of the window is not already selected, select it. Set the Label Text to 1. HRM ACTIVATION like you set the display title. Then add a polyline and move it to fit below the main step title text (but not the number). Group the line and the text by selecting them both and menu selecting Element | Grouping Group. Then set the Font to Lucida Sans Typewriter Regular (size 16) by clicking on the corresponding property and using the font picker dialog. Finally move the label to the desired location. Then save the display to a file of your convenience, e.g. 'pd_tutorial.uss'.



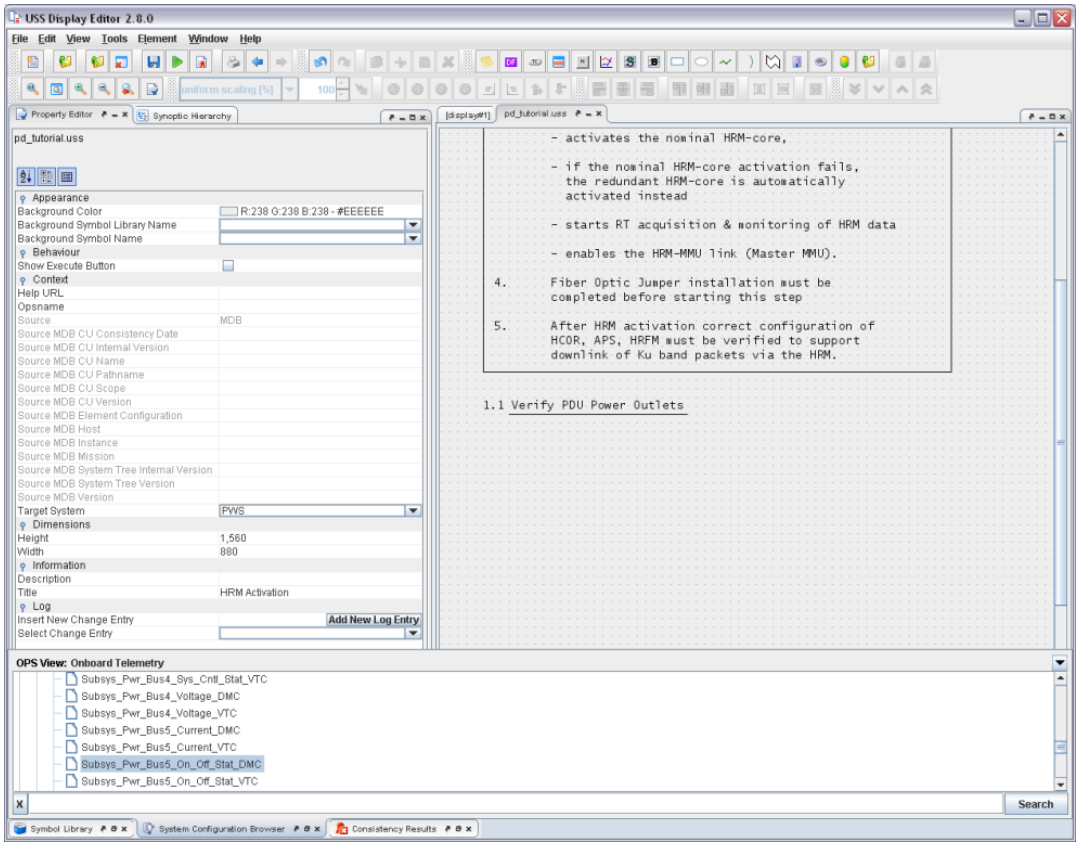
Editor after adding main step title and first saving

4. Continue by adding the note box providing clarifying information. To do this, first create the NOTE title, center the label text horizontally and underline it completely using the method already applied for the main step title above. For the note title use font Sans Typewriter Regular (size 16). Then, add note texts and numbers using font Sans Typewriter Regular (size 14) as a single multi line label It is recommended to edit the text using an external plain text editor. Note also, that the text can not be entered by means of the properties editor, as multi line labels are only supported when using in-place editing in the display area. If required, enlarge the display size to accommodate the contents. After that vertically center note title and notes and group them. Next, surround the result with a rectangle and center align the note title and text group with that rectangle both horizontally and vertically. Finally, group the note title and text group with the surrounding, aligned rectangle


USS Display Editor 2.8.0
File Edit View Tools Element Window Help
pd_tutorial.uss
Appearance
Background Color
Background Symbol Library Name
Background Symbol Name
Behaviour
Show Execute Button
Context
Help URL
Opsname
Source
Target System
PWS
Dimensions
Height
1,560
Width
880
Information
Description
Title
HRM Activation
Log
Insert New Change Entry
Add New Log Entry
Select Change Entry
1. HRM ACTIVATION
NOTE
1. The redundant 28 V power supplies from PDU1 and PDU2 (Outlet 40 on both PDU's) for the HRM are shared with other equipment (MMU, CMU, HSE, HUB). These outlets are switched ON at COL activation and remain permanently ON, independent of the HRM status.
2. For activation/deactivation, two Pwr converters in the HRM are used to supply Pwr to nominal and redundant HRM-cores (which are cold redundant) via the HRM-internal common Pwr Bus.
3. This procedure uses the FLAP: 'HRM Activation', which:
- switches ON both HRM Pwr converters,
- activates the nominal HRM-core,
- if the nominal HRM-core activation fails, the redundant HRM-core is automatically activated instead
- starts RT acquisition & monitoring of HRM data
- enables the HRM-MMU link (Master MMU).
4. Fiber Optic Jumper installation must be completed before starting this step
5. After HRM activation correct configuration of HCOR, APS, HRFM must be verified to support downlink of Ku band packets via the HRM.

Editor after adding note block

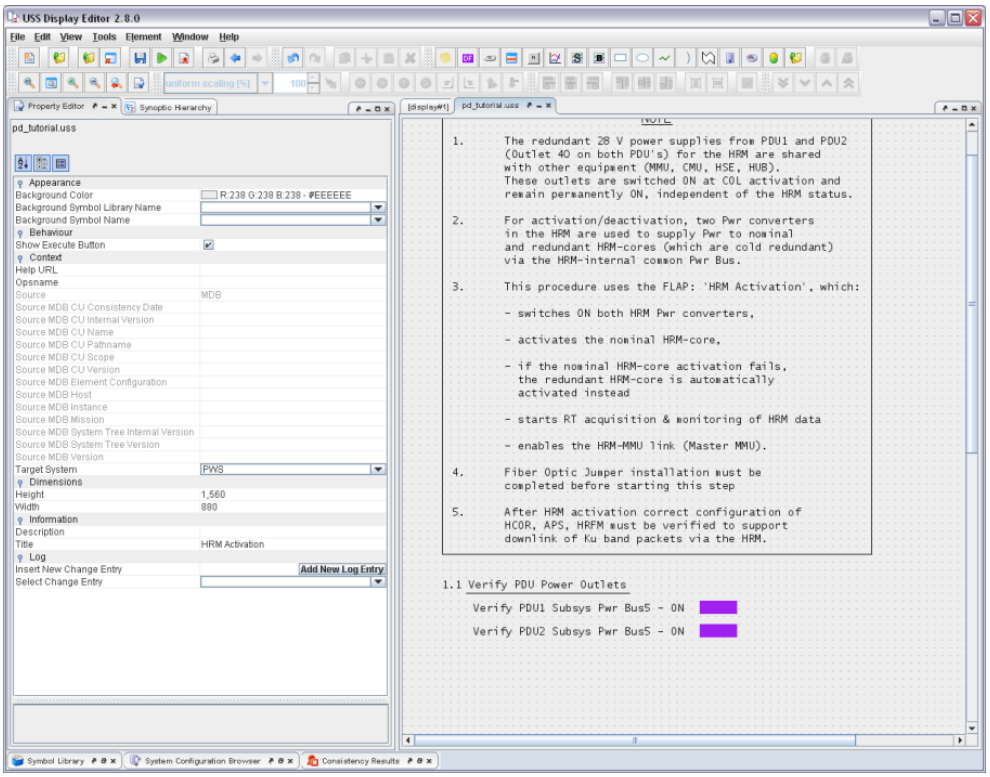
5. After adding the sub-step title 1.1 Verify PDU Power Outlets using the same method as for the main step title, but the smaller font Sans Typewriter Regular (size 14), add the corresponding verification instructions: Open the System Configuration Browser, switch to the OPS view and the Onboard Telemetry category, then browse to EPDS | PDU1 | Subsys_Pwr_Bus5_On_Off_stat_DMC.



The System Configuration Browser is shown.

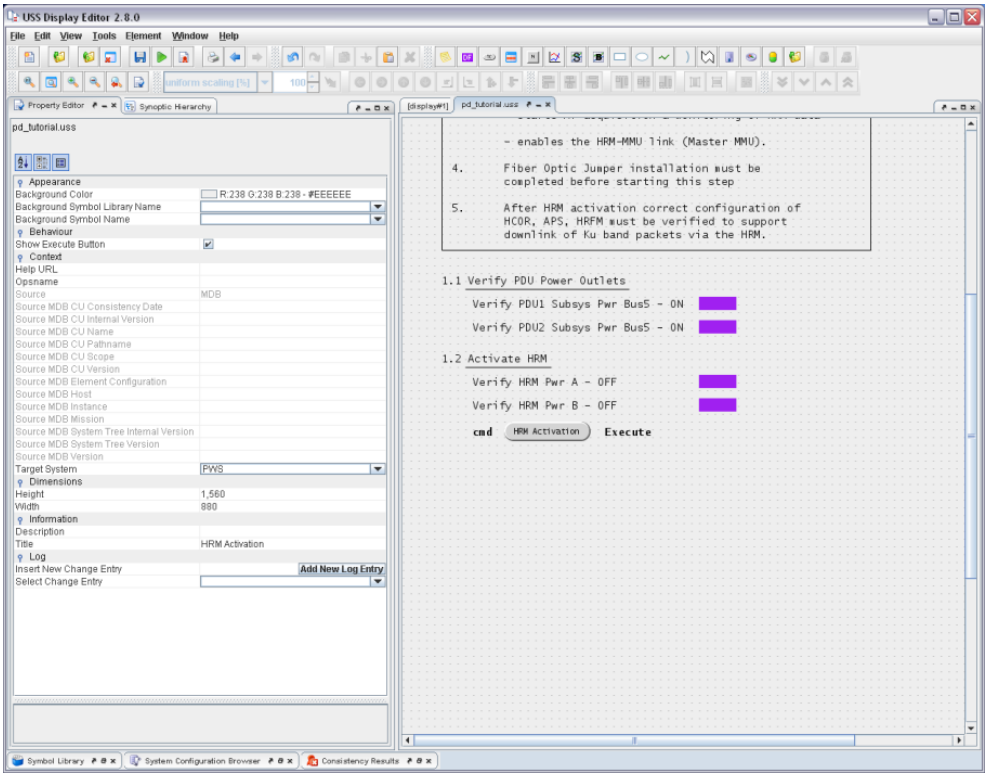
Select that item and drag it onto the display area. This will cause a label and a data filed to be

added to the display. Change the text of the label to Verify PDU1 Subsys Pwr Bus5 - ON and the font of the label as well as the data field to Lucida Sans Typewriter Sans Regular (size 14). Then align label and data field to their horizontal center. Repeat that process for EPDS | PDU2 | Subsys_Pwr_Bus5_On_Off_stat_DMC analogously. Alternatively, you can use copy and paste with manual updates to create the second verification instruction.


USS Display Editor 2.8.0
File Edit View Tools Element Window Help
Property Editor
Synoptic Hierarchy
pd_tutorial.uss
Appearance
Background Color
Background Symbol Library Name
Background Symbol Name
Behaviour
Show Execute Button
Context
Help URL
Opsname
Source
Target System
PWS
Dimensions
Height
1,560
Width
880
Information
Description
Title
HRM Activation
Log
Insert New Change Entry
Add New Log Entry
Select Change Entry
1. The redundant 28 V power supplies from PDU1 and PDU2 (Outlet 40 on both PDU's) for the HRM are shared with other equipment (MMU, CMU, HSE, HUB). These outlets are switched ON at COL activation and remain permanently ON, independent of the HRM status.
2. For activation/deactivation, two Pwr converters in the HRM are used to supply Pwr to nominal and redundant HRM-cores (which are cold redundant) via the HRM-internal common Pwr Bus.
3. This procedure uses the FLAP: 'HRM Activation', which:
- switches ON both HRM Pwr converters,
- activates the nominal HRM-core,
- if the nominal HRM-core activation fails, the redundant HRM-core is automatically activated instead
- starts RT acquisition & monitoring of HRM data
- enables the HRM-MMU link (Master MMU).
4. Fiber Optic Jumper installation must be completed before starting this step
5. After HRM activation correct configuration of HCOR, APS, HRFM must be verified to support downlink of Ku band packets via the HRM.
1.1 Verify PDU Power Outlets
Verify PDU1 Subsys Pwr Bus5 - ON
Verify PDU2 Subsys Pwr Bus5 - ON

The first sub-step is completed.

6. Now repeat the activities for adding a subs-step title and two verification instructions analogously (HRM_Pwr_Stat_A and HRM_Pwr_Stat_B can be found under DMS | HRM) for the beginning of the next sub-step. After that, add the command instruction. Switch the System Browser to the OPS view and the Onboard Commands category, then browse to COMMS | HRM | Activation_AP and drag this item onto the display area. Change the text of the created command button to HRM Activation, then add two labels for cmd and Execute and set their font to Lucida Sans Typewriter Sans Bold (size 14). Finally align the newly created components.



Editor after adding the second sub-step title, first two verification instructions and command instruction.

7. Finally continue by adding the clear text instruction to wait for a minute as a label and adding the remaining verification instructions. You can preview the resulting display by menu selecting Tools | Preview Display.

Chapter 4

Editor

4.1 Introduction

The USS Editor is the application for creating USS displays for later execution in the USS Executor. The Editor supports the definition of layout, composition and dynamic properties of synoptic displays. The operations of the Editor will be familiar to users of MS Visio with the addition of connecting graphical elements with MDB end-items.

The Editor can connect to the Columbus MDB to retrieve and store display definitions or it can work with displays directly on the file system.

The Editor comes with static and dynamic symbols as defined in Appendix C of the Display and Graphics Commonality Standard. The symbols are found in the Symbol Library and can be dragged and dropped onto the synoptic display.

The Editor can import existing display definitions from

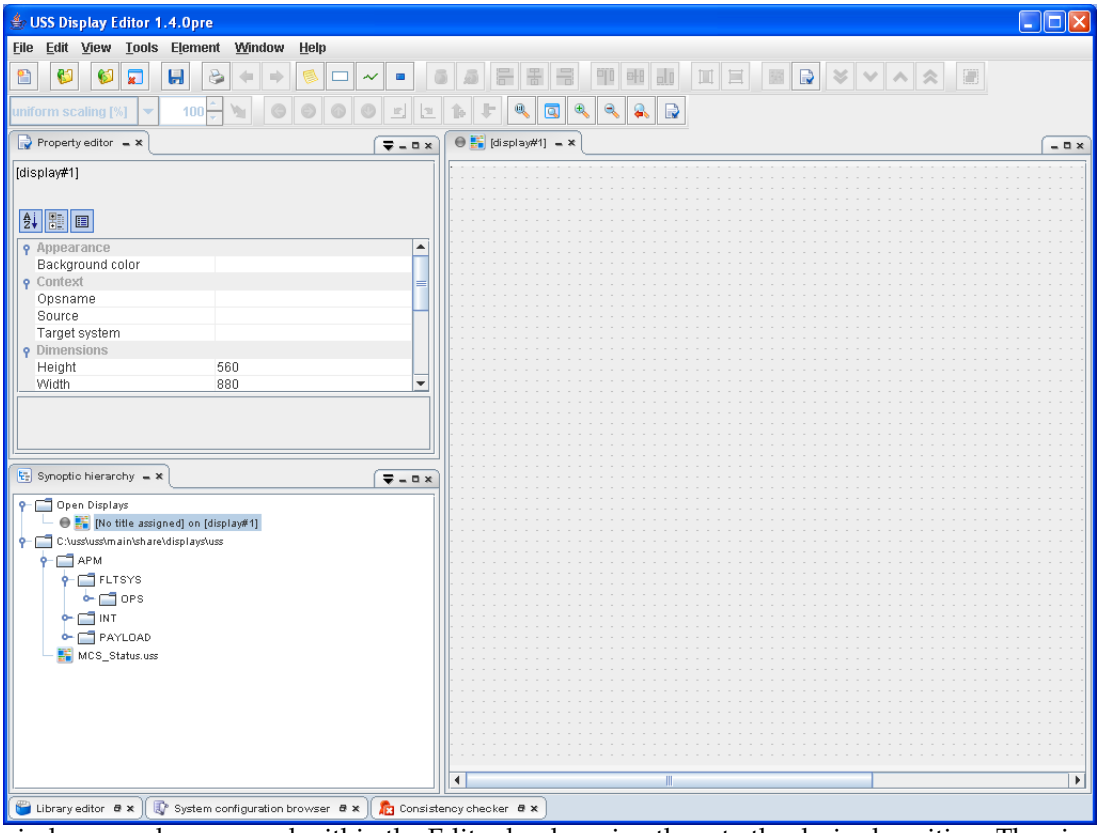
- SAMMI II
- DataView/Gipsy
- PREP

Imported display definitions become USS displays meaning that they will have the USS XML-based file format and the .uss file extension. The imported displays cannot be exported back into the legacy display formats.

4.2 The Editor Workspace

When opening the Editor, the workspace will appear. The workspace contains menus, toolbars, the display area in a tab window to the right and a number of views to the left.

The Editor menus adapt to selections and disables options that are not possible, examples are MDB and element operations



The windows can be arranged within the Editor by dragging them to the desired position. The views in the Editor are:

- The display area: The area for creating the display by adding and arranging elements
- System configuration browser: The System configuration browser is a tool-view for the Editor to ease creation of data sources, commands and navigation. It is based on the SCOE XML file in a tree structure. The System configuration browser can show the SCOE information in OPS or Path view (OPS or path -name is used for sorting SCOE xml-nodes). It provides Drag and Drop functionality for TM and TC information and filtering and hierarchical browsing of the selected MDB configuration
- Synoptic hierarchy: The Synoptic Hierarchy is a tool-view for the Editor to ease navigation of currently open displays and the USS project displays (defined by the project root-folder). It provides possibility to create navigation target to project displays, and easy opening hereof. Furthermore it provides rename, delete and explorer functionality
- Symbol library: The library of symbols that can created, edited and dragged onto the display
- Property editor: View and edit the properties of the selected element(s)
- Display View Settings: Preferences for current display
- Consistency results: Report of the last run consistency check

The toolbars can be dragged and docked to any position.

4.2.1 Arranging Views and Tab Windows

The layout of the Editor can be customized. Windows can be minimized, maximized, closed and moved around to dock in order to create a personalized window layout.

4.2.1.1 Docking mechanism of editor panels

The USS Editor features a docking framework for tool- and display views. By default views are opened as frames inside the main editor application. Small docking control icons allows you to:

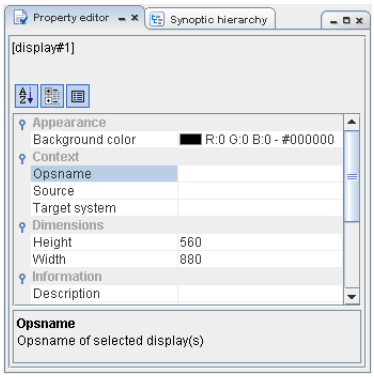
1. Undock/Minimize/Close - when view is docked, i.e. inside main window

2. Dock/Close - when view is undocked, i.e. appears in a separate window

3. Undock/Minimize/Maximize/Close - when views are collected in tabs (multiple views in tabs)

4.2.1.2 Basic Concepts

The *Views* listed above are layed out in *tab windows*.

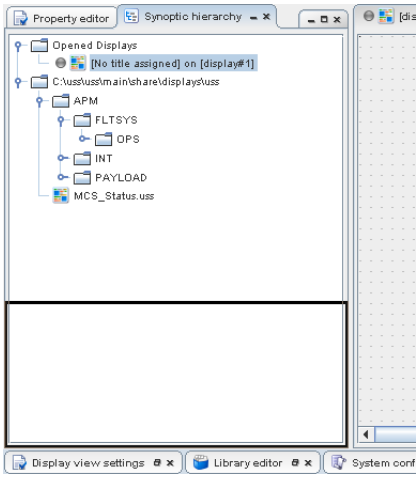


A Tab Window containing two Views: Property Editor and Synoptic Hierarchy

A *window divider* can be dragged with the left mouse button to resize the windows. A window divider is the grey area between windows. When the mouse is over the window divider it changes shape to a double arrow.

4.2.1.3 Drag and Drop Windows

Rearrange the docking windows by dragging and dropping. To move a window press and hold down the left mouse button on its tab and drag the window. A *docking frame* will show where the window will dock when releasing the mouse.



A docking frame

The drag operation can be aborted using the right mouse key or Esc.

A complete tab window can also be dragged using the area to the right of the tabs.

A view can be dragged into another tab window.

The tabs of the views within a tab window can be rearranged by dragging the tabs to their new position.

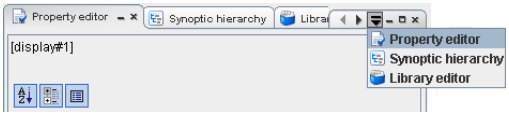
4.2.1.4 Minimize and Maximize

Views and tab windows can be minimized to the *window bar* at the bottom edge of the Editor by clicking the minimize button. The previous location of a minimized or maximized window is remembered so that it can be restored to that location. The minimized window can also be shown by clicking on it. The window can be hidden by clicking on the tab again. The windows can be restored by clicking the restore buttons. A tab window can be maximized by clicking the maximize button or by double clicking the tab.

The window bar

4.2.1.5 Tab Layout

Tabs are scrolled when there is no room for all the tabs to be visible at the same time. A tab can then be quickly selected via a drop down list. The selected tab is scrolled so it becomes visible.



Scrolling tabs

4.2.2 Editor Menus

The Editor menus adapt to selections and disables options that are not possible, examples are MDB and element operations

The following top menu items exist in the editor:

- **File**: File / Display manipulations (open, save, reload, MDB operations, print etc.) Click for File menu description
- **Edit**: Editing / selecting of displays and elements, and preferences. Click for Edit menu description
- **View**: Opening / Closing of editor tool views. Click for View menu description
- **Tools**: Editor Internal / External Tools. Click for Tools menu description
- **Element**: Display element manipulations. Click for Element menu description
- **Window**: Window zooming and navigation. Click for Window menu description
- **Help**: Help menu. Click for Help menu description

4.2.2.1 Editor Edit Menu Item

The following operations are available from the **Edit** menu (some operations are inactive if they are not feasible):

- **Undo**: Undoes the latest edit in current display
- **Redo**: Redoes the latest undo in current display
- **Copy**: Copies current element selection to clipboard
- **Cut**: Cuts current element selection to clipboard
- **Paste**: Pastes content of clipboard to current display
- **Duplicate**: Duplicates current element selection in current display
- **Set Element as Default**: Sets current selected element as default for new elements
- **Delete**: Delete current element selection
- **Button Edit**: Sub menu for button release / press look editing

- **Select Invert**: Inverts the current element selection (non-selected becomes selected and vice a versa)
- **Select All**: Selects all elements in current display
- **Select All of Same Type**: Selects all elements of same type in current display
- **Select All of Same Depth**: Selects all elements of same depth (depth property) in current display
- **Focus to Property Editor**: Opens / focuses to the Property Editor showing properties for currently selected elements
- **Preferences**: Open the preferences configuration dialog

4.2.2.2 Editor View Menu Item

The following operations are available from the **View** menu (by toggle a view is meant: If view is open it is closed, and vice a versa):

- **System Configuration Browser**: Toggles the System Configuration Browser tool view
- **Synoptic Hierarchy**: Toggles the Synoptic Hierarchy tool view
- **Symbol Library**: Toggles the Symbol Library tool view
- **Property Editor**: Toggles the Property Editor tool view
- **View Settings**: Toggles the View Settings tool view
- **Consistency Results**: Toggles the Consistency Results tool view
- **Toolbar**: Sub-menu Toolbar for adding/removing toolbars
- **Refresh**: Refresh of open views

4.2.2.3 Editor Tools Menu Item

The following operations are available from the **Tools** menu (some operations are inactive if they are not feasible):

- **Preview Display...**: Opens the Previewer for currently selected display
- **Open Display in Executor...**: Opens the Executor for currently selected display (save needed)
- **Consistency Check**: Executes a consistency check for currently selected display (opens Consistency Results on errors)
- **Run Batch Operation...**: Open the dialog for executing batch operations
- **Find Parameter...**: Open dialog for selecting an end-item to search for in project-displays (in project-root-folder)

4.2.2.4 Editor Element Menu Item

The following operations are available from the **Element** menu (some operations are inactive if they are not feasible):

- **Add**: Sub-menu containing all display elements that can be added to current selected display
- **Align**: Sub-menu containing all alignment operations that can applied to current selected elements
- **Grouping**: Sub-menu containing all grouping operations that can applied to current selected elements
- **Move**: Sub-menu containing all movement operations that can applied to current selected elements

- **Depth**: Sub-menu containing all depth arrangement operations that can applied to current selected elements
- **Rotate**: Sub-menu containing all rotation operations that can applied to current selected elements
- **Scaling**: Sub-menu containing all scaling operations that can applied to current selected elements
- **Transform**: Performs a transform to current selected elements

4.2.2.5 Editor Window Menu Item

The following operations are available from the **Window** menu (some operations are inactive if they are not feasible):

- **Zoom**: Sub-menu containing all zooming operations that can applied to current selected display
- **Back**: Goes back to previously selected display
- **Forward**: Goes forward to previously selected display (opposite direction of Back)

4.2.2.6 Editor Help Menu Item

The following operations are available from the **Help** menu (some operations are inactive if they are not feasible):

- **Help Contents**: Opens the Help contents in the web-browser
- **Search...**: Opens the a search dialog for searching the help contents
- **About...**: Shows the product information for the USS Editor

4.3 Working with Displays

The following operations are available from the **File** menu (some operations are inactive if they are not feasible):

- **New**: Create new, empty USS display
- **Open...**: Open existing USS, FWDU, GWDU or PREP display from the file system
- **Reload**: Revert to the saved version of the current display file. This function can be used to discard changes made in the display since the last save or to acquire changes made in the display file outside the editor
- **Close**: Close the current display
- **Close all**: Close all open displays
- **Save**: Save the current display to the file system
- **Save as...**: Save the new display to the file system
- **Save all**: Save all open displays to the file system
- **MDB | Browser (sync/lock-for-edit/delete)**: Show the MDB browser
- **MDB | Open from MDB**: Open display from the MDB
- **MDB | Forced Open from MDB (revert)**: Open display from the MDB and overwrite local display file, no matter synchronization state
- **MDB | Lock-for-Edit on MDB**: Lock-for-Edit the display from the MDB
- **MDB | Add to MDB**: Add a new display to the MDB
- **MDB | Save in MDB**: Save display in the MDB

- **MDB | Delete in MDB**: Delete display in the MDB
- **Create HTML Report...**: Create a detailed report about the current display
- **Export to Image...**: Export the current display to an image with or without annotation, supported formats [GIF, PNG, SVG]
- **Print...**: Print the current display
- **Print Preview...**: Preview the print of current display
- **Preferences...**: Change Editor preferences for new displays
- **Recent files**: List of the 4 most recent files for easy access
- **Exit**: Exit the Editor

4.3.1 Merging Displays

For users of GWDU the function of merging displays will be familiar. In the USS Editor merging is done by selecting, copying and pasting elements between the displays to be merged. Copy-pasting can be done as described in Basic Operations.

4.3.2 Previewing Displays

The Editor allows for previewing USS displays. From the menu select Tools | Preview... If the display has been modified since it was last saved, the Editor pops-up a dialog requesting for the display to be saved.

The preview will show the synoptic display as it will look when executed (see screen shot below). The preview mode supports different ways of entering simulation data into the display as described in the following sections.

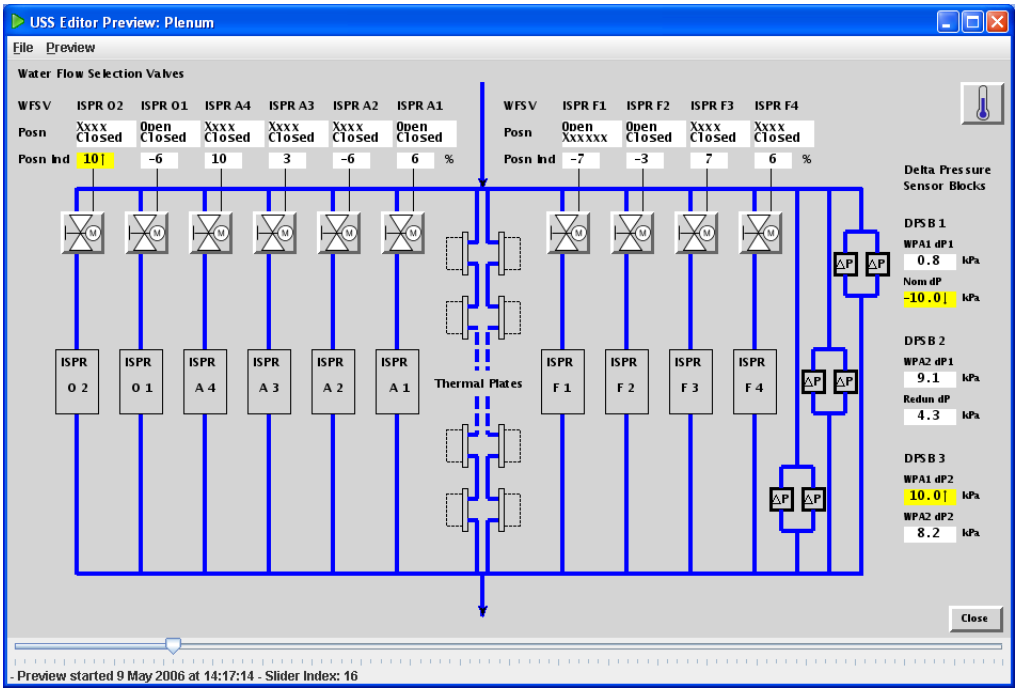
Automatically generated values used by the animator and slider are generated by a Sinus curve and also cycles through varying acquisition and monitoring states.

TIP

The automatically generated values generated by the animator and slider are generated by a sinus curve based on the range information in the configured SCOE file, if no range data exist the sinus amplitude is 10 (max. 10, min.-10)

TIP

The automatically generated values are mapped appropiately to state codes or otherwise if data source type is not numeric


USS Editor Preview: Plenum
File Preview
Water Flow Selection Valves
WFSV ISPR O2 ISPR O1 ISPR A4 ISPR A3 ISPR A2 ISPR A1
Posn Xxxx Closed Open Closed Xxxx Closed Xxxx Closed Xxxx Closed Open Closed
Posn Ind 10 -6 10 3 -6 6 %
WFSV ISPR F1 ISPR F2 ISPR F3 ISPR F4
Posn Open Xxxxxx Open Closed Xxxx Closed Xxxx Closed
Posn Ind -7 -3 7 6 %
Delta Pressure Sensor Blocks
DPSB 1
WPA1 dP1 0.8 kPa
Nom dP -10.0 kPa
DPSB 2
WPA2 dP1 9.1 kPa
Redun dP 4.3 kPa
DPSB 3
WPA1 dP2 10.0 kPa
WPA2 dP2 8.2 kPa
ISPR O 2
ISPR O 1
ISPR A 4
ISPR A 3
ISPR A 2
ISPR A 1
Thermal Plates
ISPR F 1
ISPR F 2
ISPR F 3
ISPR F 4
ΔP
Close
Preview started 9 May 2006 at 14:17:14 - Slider Index: 16

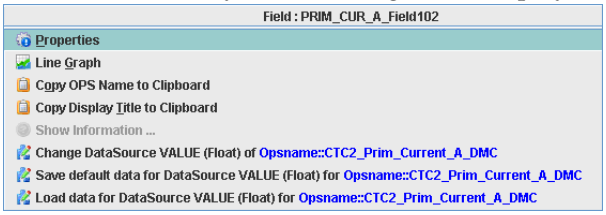
The previewer

4.3.2.1 Animator

Using the Animator (Preview | Animator) a new value is applied to each field every second.

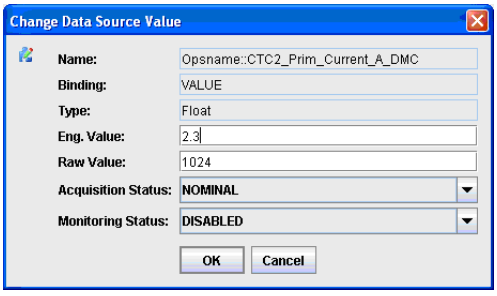
4.3.2.2 Slider

The slider applies a new value when the slider is moved to a new position. Each step on the slider has a position number and the values are always the same for a specific position. If e.g. the slider is moved to position 5 and then to position 6 and back to position 5 again then the values are the same as the first time the slider was at position 5. This consistency holds as long as the display is unchanged.


Field : PRIM_CUR_A_Field102
Properties
Line Graph
Copy OPS Name to Clipboard
Copy Display Title to Clipboard
Show Information ...
Change DataSource VALUE (Float) of Opsname::CTC2_Prim_Current_A_DMC
Save default data for DataSource VALUE (Float) for Opsname::CTC2_Prim_Current_A_DMC
Load data for DataSource VALUE (Float) for Opsname::CTC2_Prim_Current_A_DMC

The previewer context menu

4.3.2.3 Slider With Manual Specification



Manual specification of data source values.

When a DataField or another element with a data source is right clicked in Slider mode the popup menu above appears. Selecting Change data source..., the dialog above appears. This dialog allows to directly enter the value of a data source. Note that when the slider is moved, the manually entered value is lost.

4.3.2.4 Slider With Values From File

When the slider is used the data source values can be loaded from and saved to file. Save and load of data source values are available from the popup menu shown above. The layout of the data source value files is illustrated in the figure below. The files are comma separated and can be modified with a text editor.

```
Engineering Value, Raw Value, Acquisition Status, Monitoring Status
-8.66025403784439,-8.66025403784439,NOMINAL,IN_LIMITS
-7.965299180241967,-7.965299180241967,NOMINAL,DISABLED
-7.144726796328037,-7.144726796328037,NOMINAL,IN_LIMITS
-6.211477802783113,-6.211477802783113,NOMINAL,IN_LIMITS
-5.180270093731311,-5.180270093731311,NOMINAL,IN_LIMITS
-4.0673664307580095,-4.0673664307580095,NOMINAL,DISABLED
-2.8903179694447223,-2.8903179694447223,NOMINAL,IN_LIMITS
-1.6676874671610273,-1.6676874671610273,NOMINAL,IN_LIMITS
-0.4187565372920082,-0.4187565372920082,NOMINAL,IN_LIMITS
0.8367784333231448,0.8367784333231448,NOMINAL,DISABLED
2.079116908177585,2.079116908177585,NOMINAL,IN_LIMITS
3.288666467385826,3.288666467385826,NOMINAL,IN_LIMITS
4.4463517918492705,4.4463517918492705,NOMINAL,IN_LIMITS
5.533915492433431,5.533915492433431,NOMINAL,DISABLED
6.534206039901047,6.534206039901047,NOMINAL,IN_LIMITS
7.431448254773937,7.431448254773937,NOMINAL,IN_LIMITS
8.211492091337036,8.211492091337036,NOMINAL,IN_LIMITS
8.862035792312145,8.862035792312145,NOMINAL,DISABLED
9.372819894918912,9.372819894918912,NOMINAL,IN LIMITS
```

4.3.2.5 Manipulating previewed elements

Each element being previewed has a pop-up menu (mouse-right-click), which can be used to manipulate and copy the element to clipboard.

Pop-up menu items for previewed elements (not all items are present for all elements):

- **Properties**: Change properties for element, this will open a dialog with properties that can be changed in preview mode
- **LineGraph**: Shows the attached data source values in a Line Graph to for showing of variation over time
- **Copy OPS name to clipboard**: Copies the OPS name to clipboard, meaning it can be pasted elsewhere as text (other program etc.)
- **Copy Display title to clipboard**: Copies the display title to clipboard, meaning it can be pasted elsewhere as text (other program etc.)
- **Show Information**: Open a dialog showing any information available for element
- **Change Data Source Value ...**: Open a dialog for manipulating directly in preview-mode the value and states of the data source attached to the element
- **Save default data for Data Source Value ...**: Saves the samples created for the data source attached to the element, this enables manipulation of the data, so states of interest for data source can be reached
- **Load data for Data Source Value ...**: Loads previously saved samples a data source, this enables manipulation of the data, so states of interest for data source can be reached

4.3.2.6 Preview in Executor

The display can also be opened directly in the Executor by selecting Tools | Open Display in Executor... from the menu. When you select "open display in executor" a new executor instance will be started, which is not connected to a remote system, but which is running in preview mode. Any executor instance already running will not be affected.

4.3.3 Target System and DQI Style

The target system for the display is defined with the property **Target system** of the display: Click the display, open the Property Editor and locate the property Target system.

The allowed target systems and their associated DQI style file are listed in the table below.

Table 4.1: Target systems and DQI style files

Target system	DQI style file
PWS	pws_dqistyle.xml
PCS	pcs_dqistyle.xml
MCS	mcs_dqistyle.xml

The style files define foreground and background colours, characters and tooltip to display on data fields for all combinations of acquisition and monitoring status.

The style files can be edited in any standard text editor. Changes take effect when the Editor/Executor is restarted.

4.4 Working with Elements

USS displays are built up by adding and arranging elements on the display area and linking these elements to MDB end-items.

All elements provided with the USS Editor are IDAGS compatible and are:

- Label
- Data field (including LCD)
- Navigation button
- Command button
- Command list / combo-box
- Graphs (Line, Strip, and Bar graph)
- Shapes (Polyline, Rectangle, Ellipse, Arc, Polygon)
- Symbol (including Status light)
- Image
- Meters (Linear, Elliptic, Tank and Thermometer)
- Pipe, Valve and CheckValve
- Input field

See the section Elements for information about specific properties and handling of the elements.

Symbols in the Symbol Library can either be pre-defined as shipped with the USS Editor or user-defined. See Creating New Symbols to create new symbols in the library.

4.4.1 Adding and Deleting Elements

There are a number of ways to add elements to a display:

- Use the toolbar buttons **Add new label**, **Add new rectangle**, etc.
- Right-click on the display area and use the context menu that appears.
- Use the **Element | Add** menu
- Use a keyboard shortcut. Keyboard shortcuts are shown for each command in the menu of the Editor.
- Drag symbols from the Symbol Library
- Data Fields and Commands: Drag TM or TC from the System Configuration Browser
- Drag elements from the Synoptic Hierarchy Open Displays onto the current display

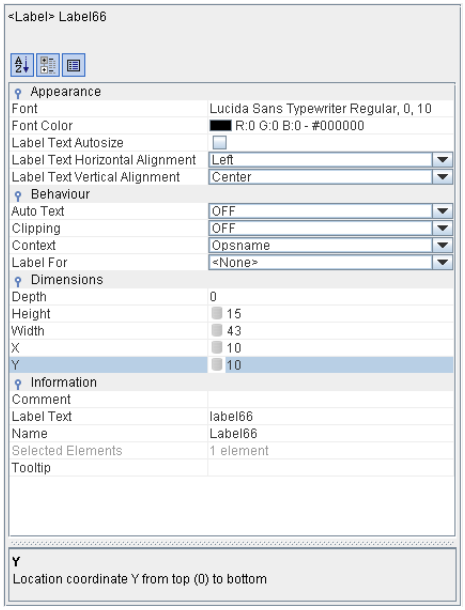
- Drag elements between display folders in the Synoptic Hierarchy Open Displays
- Navigation button: Right click the Synoptic Hierarchy and select **Add Navigation to display**

Elements are deleted by selecting the elements and:

- pressing the **Delete** button; or
- selecting **Edit | Delete**

4.4.2 Editing Element Properties

All properties of the elements are available for inspection and/or modification in the Property Editor show below.



Property Editor showing properties for a label

The top line in the Property Editor shows the element type and name. The three buttons are used for sorting the properties, toggling between category and flat list view and for toggling the bottom description pane on and off.

Categories can be folded and unfolded by clicking the handle icon next to each category title.

The Property Editor has different in-line fields for setting the properties:

- Edit field
- Drop-down box
- Check box
- Button **...** popping up a custom dialog
- Insert, Add or Remove button changing the fields of the Property Editor
- Read-only properties such as **Element type** shown their property name in grey.
- Database icon for dynamic properties.

Properties for the **Command button** element in the **Command** category are entered as a text string. Note that the Editor does not perform any check on the correctness of the command string entered.

4.4.3 Selecting Elements

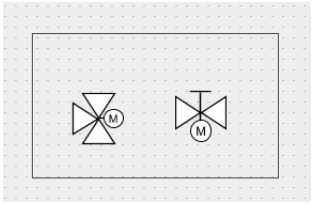
An element on a display is selected by clicking it with the left mouse button.

Use the Synoptic Hierarchy to:

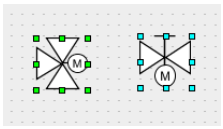
- To select an element that is covered by another element

- To select the labels on a button

Multiple elements are selected by dragging a selection rectangle around them (see figure below) or by holding down the Shift key while clicking with the left mouse button.



Two elements and a selection rectangle



Two selected elements: Primary and secondary

The *primary* selected element has green selection rectangles whereas the *secondary* selected elements have cyan selection rectangles. The align operations exploit the concept of primary and secondary selection.

Elements are deselected by clicking the display area outside the selection or selecting **Edit | Deselect**. Further selection operations available on the **Edit** menu are:

- **Select Invert**
- **Select All**
- **Select Elements of Same Type**
- **Select Elements of Same Depth**

4.4.4 Basic Operations

The Editor generally tries to position a element to have valid positions and size, when being moved, resized or likewise with mouse-actions (i.e. like mouse-move of element outside display borders). However if special settings for element value is needed, the property editor can be used to set the values (i.e. set element position outside display dimensions).

- Undo-redo: All operations performed on elements in the Editor can be undone (**Edit | Undo** or Ctrl+Z) and redone (**Edit | Redo** or Ctrl+Shift+Z).

  Note that each display has its own list of undoable operations. This implies e.g. that cut-and-paste of an element from display 1 to display 2 followed by Undo (on display 2) will only undo the paste operation, not the cut operation. To make the element reappear on display 1 Undo must also be performed there. This approach is identical to e.g. Microsoft Word when working on multiple documents.

- Move: Selected element(s) can be moved with the **Element | Move** commands or the arrow keys on the keyboard
- Drag-drop: Element(s) can be dragged and dropped between displays in the Synoptic Hierarchy Open Displays
- Copy, cut, paste: Work both on and between open displays.
- Rotate: Elements can be rotated in steps of 90 degrees
- Resize: Resizing elements is done by dragging one of the selection rectangles or directly setting the width, height, x and y properties in the property editor

4.4.5 Zooming In and Out

The following zoom operations are available on the **Windows | Zoom** menu:

- **Zoom in**: zoom in with the Zoom step size that is set in the Zoom Properties - by default 10%
- **Zoom out**: zoom out with the Zoom step size that is set in the Zoom Properties - by default 10%
- **Last Zoom**: Toggle the zoom between the current and the last zoom value
- **100% Zoom:**: Reset zoom to 100%
- **Fit to window** Fit the display area in the available space
- **View Zoom and Grid Properties**: Set the zoom properties for the current display. Zoom properties for new displays are set in the User Preferences (**Edit | Preferences...**)

4.4.6 Aligning and Distributing Elements

Elements can be aligned and distributed with the functions on the **Element | Align...** menu.

Using **Element | Align | Vertically Distributed** or **Horizontally Distributed** the selected elements are distributed so space between the elements (or the overlap in case of too little space) becomes the same.

4.4.7 Using the Grid

The grid on the display area is used to align and snap elements to the display during display development. **Element | Align | Grid properties...** sets the grid properties for the current display. Grid properties for new displays are set in the User Preferences (**Edit | Preferences...**)

Example: The grid and alignment functions can be used to arrange fields in a tabular layout. In View Settings, set for example the Spacing Vertical to 2 pixels more than the height of the fields to be arranged. Then drag the fields to their approximate positions and they will snap to the correct place as illustrated in the screen dump below.

Tabular layout of fields

4.4.8 Grouping Elements

Elements can be grouped to create 'compound' elements. Compound elements can consist of any type of elements including other compounds. When elements are grouped a new compound element is automatically created. The compound element can be modified in the property editor and its constituents can be inspected in the Synoptic Hierarchy Open Displays.

4.4.9 Working with Depth

The 'depth' property of elements determines which elements are drawn on top if the elements overlap. The depth is set through the **Element | Depth...** menu or directly in the **Depth** property using the property editor. The element with the highest depth value is drawn on top.

Graphs are always drawn on top of other elements.

4.4.10 Changing the Element Default Values

All elements are created with the default element properties. The default properties for an element can be set by right-clicking on the element and selecting **Set as Default**. Affected element properties are listed in the table below.

Note that if an element is created when another element is selected, then the new element will get the size of the selected element and not the set default size.

Table 4.2: Element Properties with "Default Capability"

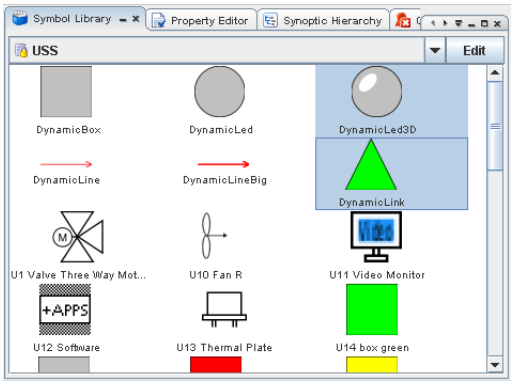
Property	Saved as default for
FillStyle (Fill colour and style)	Shapes, Buttons, Meters
TextStyle (Font face and colour)	Label, ComboBox, CommandList, Field, InputField
Rotation	Label, Buttons, Field
DrawStyle	Shapes
AutoSize	Label
AutoWrap, ShowUnit, ShowIndicators, UnitTextStyle	Field
BorderColor, Height, Width	Meters
TickIndicator, LabelStyle, FieldStyle, Fonts	Elliptic/LinearTickMeter, Thermometer
StartAngle, SweepAngle, DrawAsCircle	EllipticTickMeter
Orientation	Graph and Meters
SelectionColor	CommandList

Shapes are Arc, Rectangle, Polygon and Ellipse. Meters are TankMeter, Elliptic/LinearTickMeter and Thermometer.

4.5 Using The Symbol Library

The Symbol Library contains a number of pre-defined IDAGS compliant symbols in categories: 3D Graphics, Communication, Communications, Computers, Electrical, Events, Fluid, General, Groups, Logic, Mechanical, Misc, Modes, Radiation Monitoring, Robotics, Sensors, Station Modules, Tasks and USS.

Symbols can be dragged onto the display by clicking the symbol, holding down the right mouse button, dragging the symbol to the display and releasing the right mouse button. Note that multiple symbols can be selected by holding down the **Shift** key during selection.

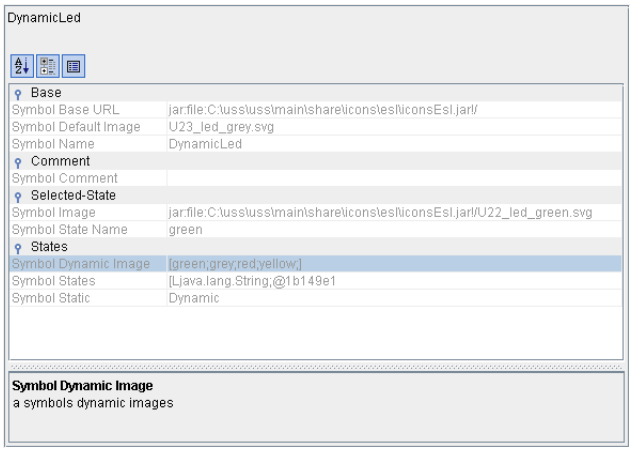


The USS library in the Symbol Library

4.5.1 Pre-Defined Dynamic Symbols

The USS library in the Symbol Library contains, among others, a number of dynamic symbols. The USS library is illustrated above.

State names for predefined dynamic symbols in the USS library can be inspected by selecting the symbol in the library and then opening the Property editor. This is illustrated below.



Inspecting properties of a symbol in the library

4.5.2 Creating New Libraries and Symbols

New 'user' libraries can hold symbols with different contents: Image or USS sub-display. **Images** can be raster images (gif, jpg or png) or vector graphics (svg). Images can be created outside the Editor in an image manipulation program, stored to file and added to a library in the USS Editor. Images can also be created inside the USS Editor but only based on already existing symbols and elements.

USS sub-displays are different from images as they are composed of USS elements whose properties are retained from their creation. USS sub-displays can be seen as synoptic templates. USS sub-displays can be used by reference or by copy. If used **by reference** the contents of the **instance**[1] will be controlled by the symbol in the library and cannot be updated in the display. If used **by copy** the instance is not connected to the symbol in the library but behaves as an ordinary group of elements.

Another aspect of a symbol is whether it is static or dynamic. A **static** symbol contains one image or sub-display whereas a dynamic can contain many images (but not sub-displays) each identified with a State Name. An instantiated dynamic symbol must be connected to a data source having the defined state names as output. At run time, the symbol will change between its states showing the associated images, depending on data source output.

Libraries are created, renamed and deleted by clicking the Edit button next to the library name, and selecting the appropriate action from the context menu that appears. Note that renaming and deleting libraries or symbols will lead to dangling references in displays that contain symbols from the renamed or deleted library. Dangling references can be found by the Consistency Checker.

A symbol in a library is created by selecting elements on the display area and clicking the Edit button next to the library name, and selecting Add Selected Elements as Symbol from the context menu that appears. After specifying the symbol name and type (image or sub-display, see above) the new symbol will appear in the Symbol Library.

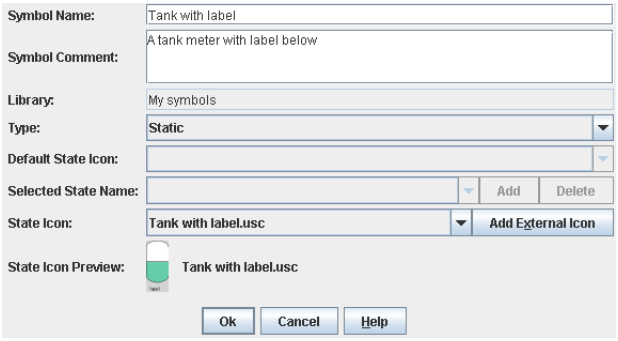
New symbols can be used by dragging them to the display as any other symbol shipped with the Editor. By dragging a sub-display to the display area a instance by reference is created. To use the sub-display by value, right click the symbol and select Add Symbol Components to Display.

It is recommended to use the Symbol Library for images on the display instead of the image element as described in section Image.

4.5.2.1 Advanced Editing of Symbols

Specifying dynamic properties of a symbol is done in the dialog below. Right click the symbol and select Edit Symbol in order to perform advanced editing.

[1] The symbol on the display as opposed to the symbol in the library



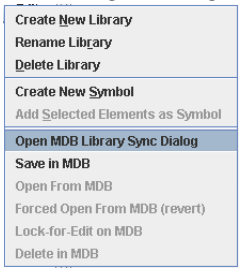
Dialog for definition of a symbol

The Symbol Comment field is for notes about the symbol, it will not be visible at run time.
To make a dynamic symbol from a static one:

1. Switch the Type to Dynamic
2. For each state:
   (a) Click Add and enter the state name
   (b) A new image for the state can be added directly by clicking Add External Icon. Sub-panels cannot be used as state icons.
   (c) Select the icon (i.e. symbol) from the State Icon list
3. Select the default state icon

4.5.2.2 MDB Version Control of Symbol Libraries

Like the USS displays the user libraries can be version controlled. The MDB interface for symbol libraries in the USS Editor, handles synchronization, submitting, locking, deleting with configured MDB.

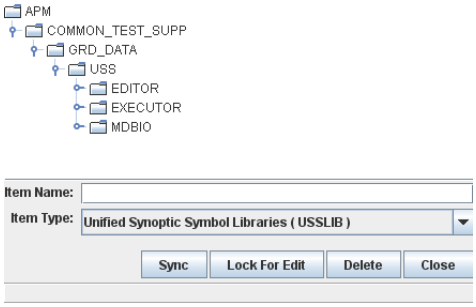


Location of MDB actions for user/MDB symbol libraries

The symbol library MDB actions are located in the symbol library view, under the edit popup-menu (press button: Edit).
MDB actions available on user/MDB symbol libraries:

1. Synchronization Browser - Browser for viewing available symbol libraries in MDB
2. Save in MDB - submit symbol library, new or modified
3. Open from MDB - synchronizing of symbol library
4. Forced Open from MDB (revert) - synchronizing of symbol library, overwriting existing local library
5. Lock-for-Edit on MDB - Lock-for-Edit of symbol library, lock library in MDB, to enable editing
6. Delete in MDB - Deletion of symbol library

MDB Library Sync Dialog is a small tool that enables the synchronization of symbol library not present in the uss-project folder. The dialog opens a connection to the MDB and gives a tree view of the current MDB tree. The MDB connection is configurable via the Preferences.



MDB Library Sync Dialog

NB! The Display MDB Browser, located under editor menu: 'File | MDB | MDB Browser' cannot be used to handle symbol library MDB actions.

TIP

In USS symbol are collected in libraries, which can be kept in the MDB or distributed with USS. The symbol libraries are references locally

TIP

The symbol libraries should be kept up-to-date locally by keeping them read-only, and having them synchronized/update from MDB, when needed, by a System administrator

4.6 Elements' Advanced Properties

This section describes specific properties and handling of the elements. The purpose is not to describe all elements types and their properties, but rather to answer frequently asked questions. Please refer to Working with elements for a complete list of the element types.

4.6.1 The Display

The **Target System** is described in section Target system and DQI style.

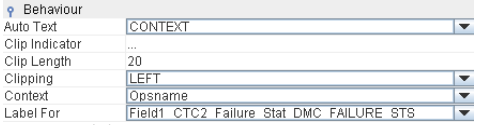
The **Display Description** appears in the Display Report. The Display Description can be HTML as shown in the screen shot below.

4.6.2 Label

The label text can be edited directly on the display by double click the label. The changes are accepted by pressing Enter and reverted by pressing Escape.

When TM is dragged onto a display, a label and data field is created. The label is associated with the field. This association/binding can be changed with the Behaviour | Label For property of the label.

When the label is associated with a field the properties in the screen shot below can be used to change the text that the label shows and clipping of the text.



The Auto Text property has three possible settings:

- OFF: The label text is not taken from the TM.

- CONTEXT: The label text is taken from the TM as specified with the Context property.
- BASENAME: The label text is taken from the TM as specified with the Context property. This setting is different from CONTEXT only if the Context property is set to Pathname.

For the CONTEXT and BASENAME settings, note that the label text from the TM is copied into the Label Text property at editing time and not updated dynamically at run time. Further, if the SCOE file is changed the label can be inconsistent with the SCOE TM value - run a consistency check.

The Context property determines which part of the TM is copied in to the Label Text. The Context drop down list contains the contexts defined in the Editor Preferences.

Clipping determines which part of the Label Text will be shown if the label is too small to contain all the label text.

4.6.3 Data Field

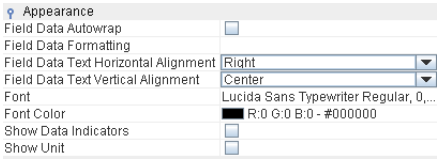
Data field can be created by dragging one or more TM from the System Configuration Browser on to an empty area on the display background.

The **Field data formatting** is described in section Field data formatting below.

The **Data Source** is described in section Data Sources below.

LCD elements are Data Fields using an LCD font. In the Property Editor, Appearance category, Font property open the "Pick a font" dialog and select font LCD or LCD2.

Data Fields properties



Data Field Properties in Property Editor

The Data Field has several unique properties:

- Data Auto Wrap: If checked the data field will wrap (put multiple lines) data which is too wide for the field
- Data Formatting: See below
- Data field width:
  - Data filed width can be changed by following properties. The width, number of columns, font and the font size. The data field width will always be overwritten if the width is too small for displaying the defined number of columns depending on defined font and font size. **It is recomended to change data field width using property "number of columns" not property "width".**
- Data Text Horizontal Alignment: Left, Center or Right alignment of data text in field
- Data Text Vertical Alignment: Bottom, Center or Top alignment of data text in field
- Font: The font to use for text
- Font Colour: The colour to use for text
- Show Data Indicators: if checked the data field will indicate quality of incoming data
- Override DQI: if checked the data field will not show use DQI (data quality ind.) colouring of the data field
- Overflow behaviour: Defines how the field should behave when data value are out-of-range:
  - OVERWRITE - Old USS behavior, never truncates values, may overwrite adjacent labels, may cause flicker, default for new fields. for new fields.
  - EXPAND - Expands field size on overflow (never shrinks), workaround for GWDU import problems where fields have too small bounding box, default for GWDU importer.
  - SHOW_HASHES - Fills field with hashes, like ''####', PCS behavior, Excel behavior, default for PCS importer.
- Show Unit: if checked the data field show unit of attached data source

4.6.3.1 Field Data Formatting

Data fields have the property "Field data formatting" which specifies the formatting of the data value in the field. The format is specified using the "printf style" which will be well-known to C programmers.

The format specifier has the following syntax:

```
%[flags][width][.precision]conversion
```

The optional flags is a set of characters that modify the output format. The set of valid flags depends on the conversion.

The optional width is a non-negative decimal integer indicating the minimum number of characters to be written to the output.

The optional precision is a non-negative decimal integer usually used to restrict the number of characters. The specific behaviour depends on the conversion.

The required conversion is a character indicating how the argument should be formatted. The set of valid conversions for a given argument depends on the argument's data type.

An empty of string flag will treat the data as a string

Conversions

Table 4.3: Conversions

Conversion	Argument Category	Description
'd'	integral	The result is formatted as a decimal integer
'o'	integral	The result is formatted as an octal integer
'x', 'X'	integral	The result is formatted as a hexadecimal integer
'e', 'E'	floating point	The result is formatted as a decimal number in computerized scientific notation
'f'	floating point	The result is formatted as a decimal number
'g', 'G'	floating point	The result is formatted using computerized scientific notation or decimal format, depending on the precision and the value after rounding.
'a', 'A'	floating point	The result is formatted as a hexadecimal floating-point number with a significand and an exponent
's'	string	The result is treated as a string, and is not formatted (String, Statecode, etc.)

Flags

The following table summarizes the supported flags. 'y' means the flag is supported for the indicated argument types.

Table 4.4: Flags

Flag	Integral	Floating Point	Description
'-'	y	y	The result will be left-justified
'#'	y [3]	y	The result should use a conversion-dependent alternate form
'+'	y [4]	y	The result will always include a sign
' '	y [4]	y	The result will include a leading space for positive values
'0'	y	y	The result will be zero-padded
','	y [2]	y [5]	The result will include locale-specific grouping separators
'('	y [4]	y [5]	The result will enclose negative numbers in parentheses

[2] For 'd' conversion only.

[3] For 'o', 'x', and 'X' conversions only.

[4] For 'd', 'o', 'x', and 'X' conversions applied to BigInteger or 'd' applied to byte, Byte, short, Short, int and Integer, long, and Long.

[5] For 'e', 'E', 'f', 'g', and 'G' conversions only.

Width

The width is the minimum number of characters to be written to the output.

Precision

For the floating-point conversions 'e', 'E', and 'f' the precision is the number of digits after the decimal separator. If the conversion is 'g' or 'G', then the precision is the total number of digits in the resulting magnitude after rounding. If the conversion is 'a' or 'A', then the precision must not be specified.

For integral types, the precision is not applicable.

Examples

Table 4.5: Examples

Syntax	Example output
%d	13
%03d	013
%5.2f	12.67

4.6.4 Command Button

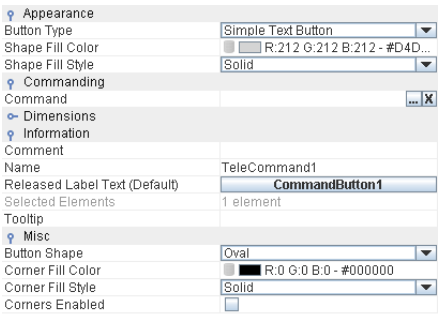
A command button can easily be created by dragging TeleCommand from the System Configuration Browser to the display area.

TIP

TeleCommands in USS can be any of the following types:

- HLCL
- SWOP
- FLAP
- PCS
- USS
- TC

Command Button properties



Command Button Properties in Property Editor

The Command Button has several unique properties

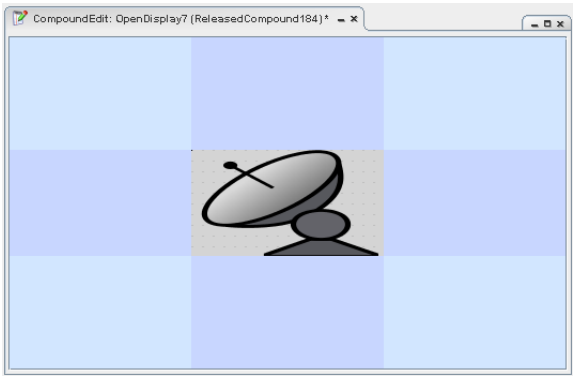
- Button Type: Defines capabilities and appearance of the button

- Shape Fill Color: Defines the fill colour of the button, i.e. the colour of the button
- Shape Fill Style: Defines the fill style, i.e. solid = fill button, none = no colouring of button
- Command: Shown attached command, press mini-button: '...' to open command editor and 'X' to remove command
- Button Shape: How button is drawn (Oval, rectangular or flat, i.e. no 3D effect)
- Corner Fill Color: Colour of drawn corners
- Corner Fill Style: Defines the fill style, i.e. solid = fill corners, none = no colouring of corners
- Corners Enabled: if checked the corners are drawn

The Button Type property aggregates two properties of a button:

- Whether it has both a pressed and a released state or is 'simple' with only one state. The state determines which label or compound is shown on the button at run-time.
- Whether the contents is Text or a Compound containing any composition of elements

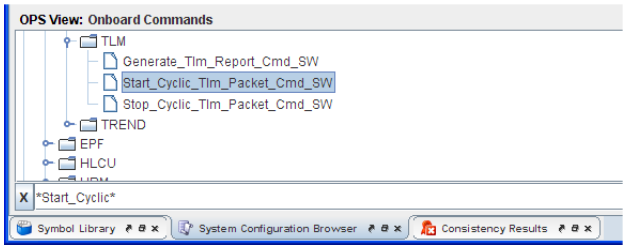
The contents of compound buttons is edited in a special mode of the Editor illustrated in the screen shot below. First set the button to the needed size. This is important as later resizing of the button will also resize its contents. Right-click it and select Edit Released/Pressed Button Compound. In the button-edit mode only a subset of the element types are available. When finished editing, close the CompoundEdit tab to return to normal mode.



The button editor

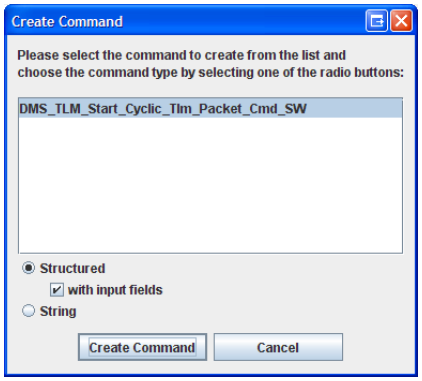
Creating a Command Button by selecting from a list of defined commands:

- Open System Configuration Browser.
- Select OPS View: Onboard Commands.



Select onboard command

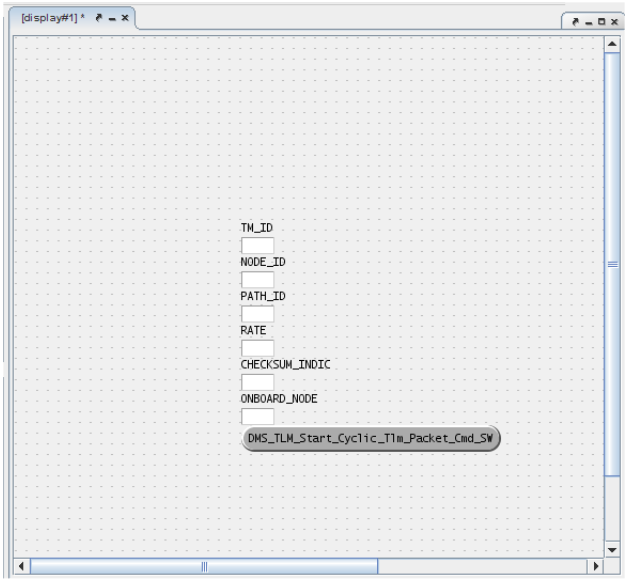
- Navigate to the command you want to attach to a command button. You may also use the search function.
- Drag and drop command to display area. The 'Create Command' window opens.



Create command button with selected command

- Select 'Structured' and 'with input fields'.

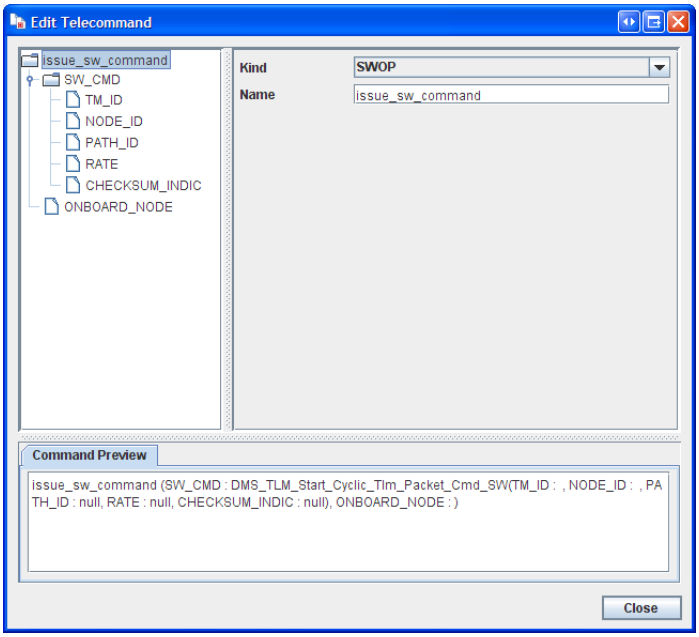
- Press 'Create Command' button. A command button and, if applicable, a set of input fields are created in the display area. Move button and fields where they shall be placed.



Command button with associated input fields

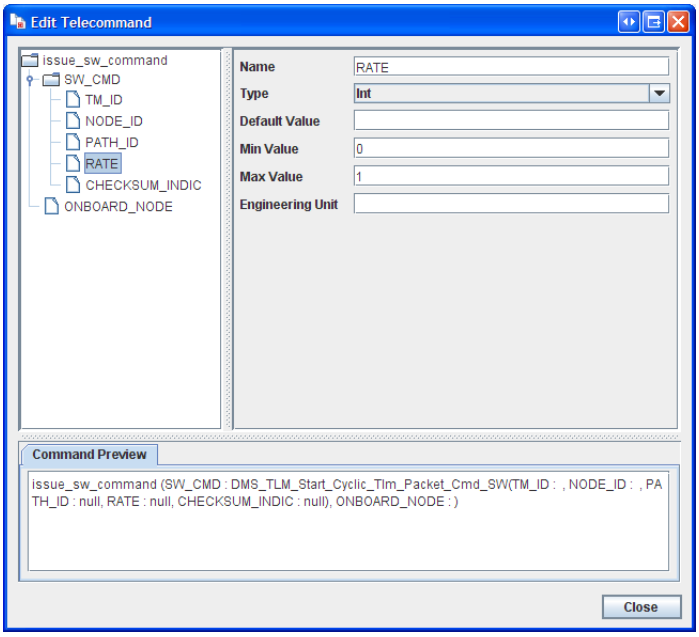
- You may now edit the properties of the Command Button and of the input fields. Start by selecting the Command Button.

- In the property editor press the mini-button '...' to open the command editor.



Telecommand editor with command preview

- Select one of the command parameters on the left side. The input field properties are shown.
- Edit parameter properties.



Telecommand editor: Edit input field

NOTE: In case you need to update the constraints of a parameter take care to set values that do not contradict to the original command definition.

The following table lists the different possible constraints of input fields.

Table 4.6: List of input field constraints

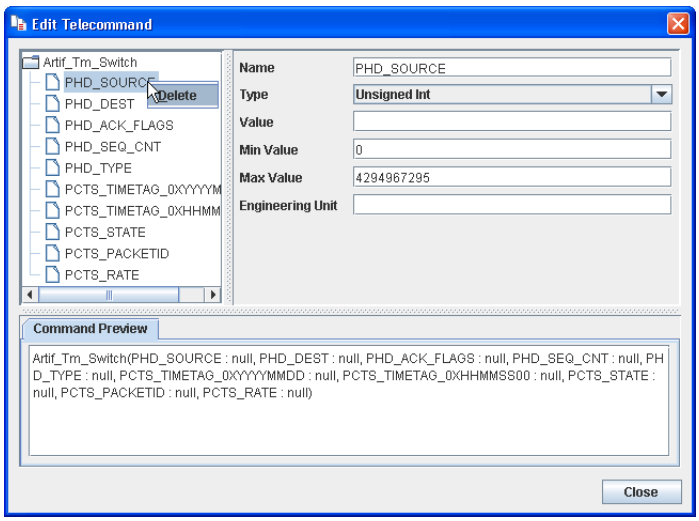
Data Type	**Valid Constraint**
Absolute Time	no constraint defined
Binary	no constraint defined
Bitset	List of integer values in the range of 0 .. 31
Boolean	'true' or 'false'
Double Float	Range definition of min. and max. value
Float	Range definition of min. and max. value

Table 4.6: (continued)

Data Type	Valid Constraint
Hexadecimal	value is a valid hexadecimal value, max. number of digits
Integer	Range definition of min. and max. value
Long Integer	Range definition of min. and max. value
Object Identifier	Either a set of valid end item types or a set of valid end item opsnames
Short Integer	Range definition of min. and max. value
State Code	Set of valid values
String	Set of valid values, or a regular expression to compare with, or max. lenght of string
Unsigned Integer	Range definition of min. and max. value
Unsigned Long Integer	Range definition of min. and max. value
Unsigned Short Integer	Range definition of min. and max. value
Word	no constraint defined

A command for a button can also be edited as follows:

- In the propertyeditor press the mini-button '...' to open the command editor.
- If a command for the button has not been defined yet, a dialog lets you choose between string telecommands and structured telecommands.
- Choose structured telecommand, if the telecommand has nested parameters, or parameters that need user input.
- The structured command editor shows a tree view of your command structure on the left hand side, a panel to modify properties of a selected parameter on the right hand side and a preview field on the bottom showing a canonical version of the produced command string.
- Following actions depending on the node type can be taken by right-clicking a node in the tree view:
  - Add Simple: Adds a simple parameter to this node that can not be further nested.
  - Add Nested: Adds a nested parameter to this node that can have further simple or nested parameters as input.
  - Delete: Deletes a parameter node from its parent.
- On the right panel side a name and a type can be chosen for each parameter node.
- For the top command node a kind and a name can be selected.

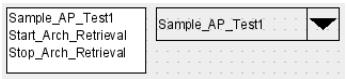


The command editor

4.6.5 Command List

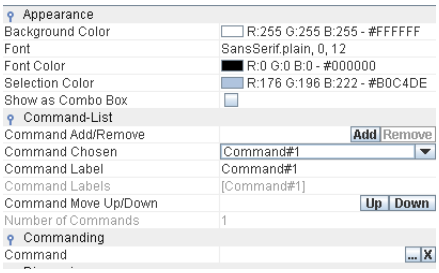
Commands can easily be added to an existing command list by dragging TC from the System Configuration Browser to the command list. Multiple commands can be dragged by selecting them in the System Configuration Browser and dragging to the command list with the *right* mouse button pressed down.

A command list is changed into a command combo box (command pop-up) with the property Show as Combo Box.



The same Command List element configured in two different ways

Command List properties



Command List Properties in Property Editor

The Command List has several unique properties (see)

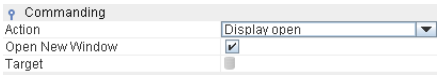
- Background Color: The color of the background inside the command list
- Font: The font to use for text
- Font Color: The colour to use for text
- Selection Color: The colour to use for selected text
- Show as Combo Box: if checked the command list is drawn as a combo-box, instead of a open list
- Command Add/Remove: Press mini-button: 'Add' to add a new command, and mini-button: 'Remove' to remove selected command
- Command Chosen: Selected command, use drop-down combo-box to select another
- Command Label: Command label, type in field to change
- Command Move Up/Down: Press mini-buttons: 'Up' and 'Down' to move the command, up or down respectively in list
- Command: Shown selected command, press mini-button: '...' to open command editor and 'X' to remove command

4.6.6 Navigation Button

The target display for a navigation button can be set by right clicking a USS display in the Synoptic Hierarchy and selecting **Add Navigation to display**

Button Type property: see Command Button.

Navigation Button properties



Navigation Button Properties in Property Editor

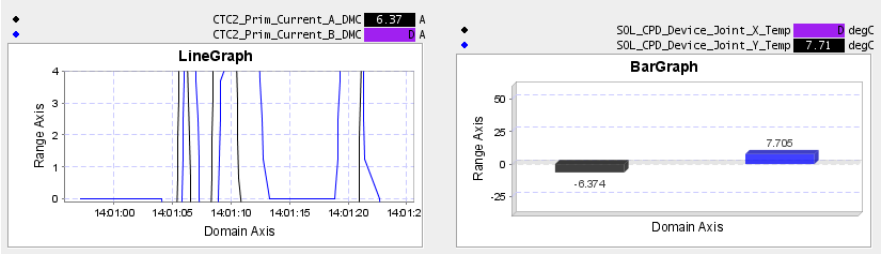
The Navigation Button has some unique properties, but share many properties with command button above

- Action: Whether the navigation target specified should be opened or closed
- Open New Window: If checked the open action will not replace the current display, but open a new display inside the USS Executor
- Target: The navigation target to apply the above action, i.e. open or close

4.6.7 Graphs

Data sources can be added to graphs by dragging TM from the System Configuration Browser.

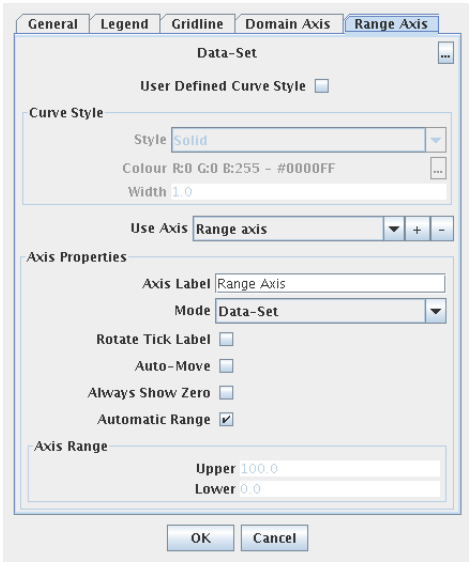
Line and Bar graph examples (Previewed in Editor)



4.6.7.1 Line Graphs

Line Graphs are configured in the Property Editor by clicking the ... button at the property 'Configure Graph'. The Line Graph Properties dialog below is opened.

Line Graph Range properties



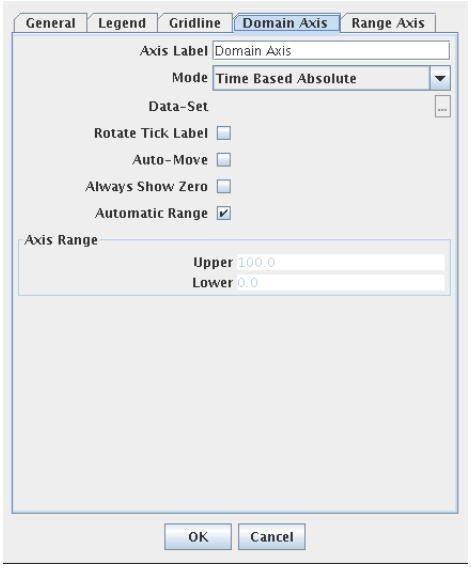
Line Graph Range properties in Graph dialog

Line Graph Range Tab properties

- Data-set: Click button: '...' to edit range data sources
- User Defined Curve Style: If checked, curve styles can be defined individually for each curve, i.e. range data source
- Curve Style (Box): If using user defined curve styles; Style: solid, dashed and dotted, colour and line-width can be defined for selected range data source (see Data-set)
- Use Axis: Select which Axis to use for selected range data source (see Data-set). Or press buttons: '+' or '-' to add respectively remove axes
- Axis Label: The label of the axis
- Mode: Select mode of selected axis (see Use Axis); select between data-set (i.e. values from range data source) or Time-Based-Absolute for time values
- Rotate Tick Label: If checked, Tick label is rotated
- Auto-Move: If checked, Graph will move automatically
- Always show Zero: If checked, Graph will always show the zero on the axis
- Automatic Range: If checked, Graph will change the range of the axis to keep values

- Upper Range: Upper limit of axis (Automatic Range unchecked)
- Lower Range: Lower limit of axis (Automatic Range unchecked)

Line Graph Domain properties

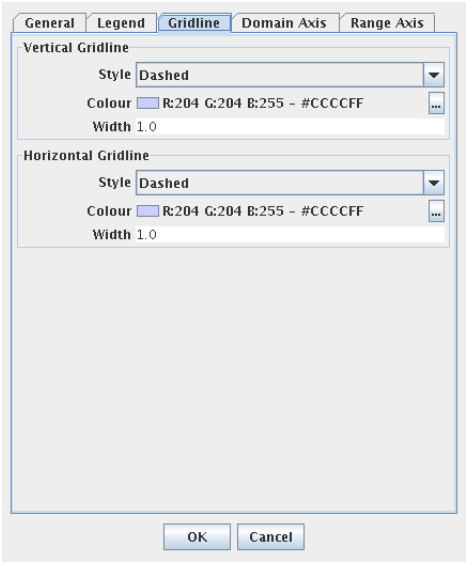


Line Graph Domain properties in Graph dialog

Line Graph Domain Tab properties

- Axis Label: The label of the axis
- Mode: Select mode of axis; select between data-set (i.e. values from domain data source) or Time-Based-Absolute for time values
- Data-set: Click button: '...' to edit domain data source (Mode set to Data-set)
- Rotate Tick Label: If checked, Tick label is rotated
- Auto-Move: If checked, Graph will move automatically
- Always show Zero: If checked, Graph will always show the zero on the axis
- Automatic Range: If checked, Graph will change the range of the axis to keep values
- Upper Range: Upper limit of axis (Automatic Range unchecked)
- Lower Range: Lower limit of axis (Automatic Range unchecked)

Line Graph Gridline properties

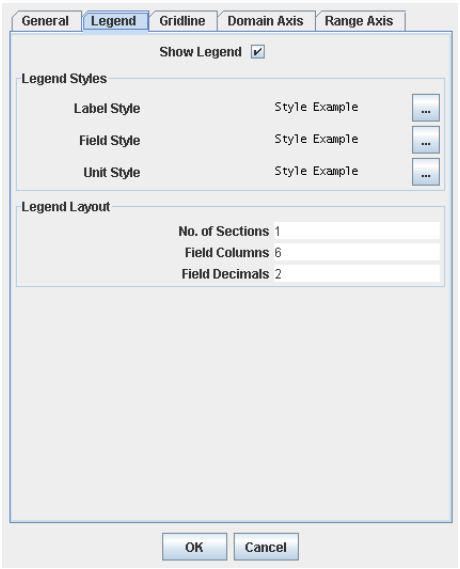


Line Graph Gridline properties in Graph dialog

Line Graph Gridline Tab properties

- Vertical Gridline (Box): Style: solid, dashed and dotted, colour and line-width can be defined for vertical gridlines
- Horizontal Gridline (Box): Style: solid, dashed and dotted, colour and line-width can be defined for horizontal gridlines

Line Graph Legend properties

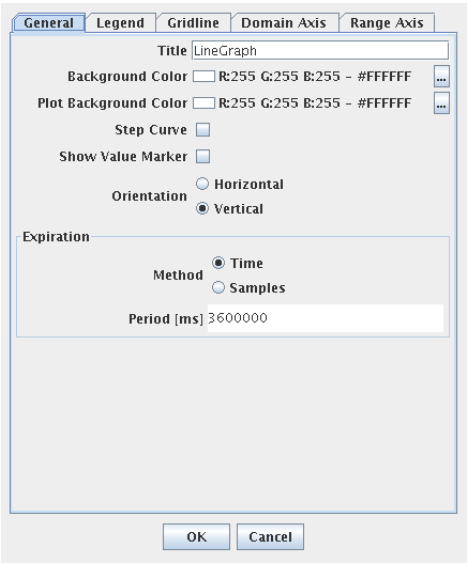


Line Graph Legend properties in Graph dialog

Line Graph Legend Tab properties

- Show Legend: If checked, legend will be shown/drawn
- Legend Styles (Box): Click buttons: '...' to edit label, field and unit text styles respectively
- Label Color: Click button: '...' to edit label colour in legend
- No. of Sections: Sets number of section the legend fields are put into
- Field Columns: Defines how many columns the fields uses, i.e. how many numbers
- Field Decimals: Of the columns above, field decimals defines how many of them are used for decimal part of field value

Line Graph General properties



Line Graph General properties in Graph dialog

Line Graph General Tab properties

- Title: Title of graph in display
- Background Color: Defines background colour of graph
- Plot Background Color: Defines plot background colour of graph, i.e. background of curves drawing area
- Step Curve: If checked, graph are drawn in step, i.e. non-continuously
- Show Value Marker: If checked, graph places a marker for every real value, i.e. to indicate points that are not interpolated
- Orientation: Choose between Horizontal and vertical orientation of plot inside graph
- Expiration (Box): Choose method for defining expiration, choose between Time (how long time to keep samples/values) and samples (how many samples/values to keep). And define in input field below the actual value.

A graph can be converted into a real-time graph by setting the check mark the **LineGraph range axis auto move** property. A real-time graph has a time-based horizontal axis where the plot continues and scrolls to the left as times passes on even when no parameter update is received.

4.6.7.2 Strip Graphs

The property **StripGraph LineGraph Weight** can be used to set the relative height og the line graphs within the strip graph. The property has no unit -- the relative height is set as the weight relative to the total weight for all line graphs in the strip graph.

Strip Graph properties

Strip Graph Properties in Property Editor

The Strip Graph element has several unique properties.

- Configure Graph: Click mini-button: '...' to edit main strip graph (opening dialog with a sub-set of line graph properties)
- Select Sub Graph: Control sub graphs (line graphs), change ordering with up/down arrows, select sub graph with drop-down combo box, and press mini-button: '...' to edit currently select sub graph (opening dialog with a sub-set of line graph properties)
- Sub Graph Weight: Set the currently select sub graph weight, see above for weighting explanation

4.6.7.3 Bar Graphs

The Bar Graph element is a chart graph / histogram.

The Bar Graph has many common properties with the line graph, containing more or less as sub-set of line graph properties.

Bar Graph Range Tab (differences compared to line graph)

- Contains only one range axis definition common for all range data source
- No Tick and therefore no property: Rotate Tick Label
- No Auto-Move property
- Bar Graph has the possibility to limit the automatic range, properties: Automatic Range Lower/Upper Bound

Bar Graph Domain Tab (differences compared to line graph)

- Bar Graph does not have any domain data sources or axes, therefore only domain axis property: Label

Bar Graph General Tab (differences compared to line graph)

- Bar Graph does not have value markers and step curve properties

4.6.8 Arc

The Arc element is drawn as a sub-section of an ellipse, where the size of the element determines the shape of the ellipse.

Arc properties

Appearance	
Arc Angle Length	180
Arc Start Angle	0

Arc Properties in Property Editor

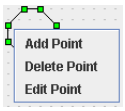
The Navigation Button has some unique properties

- Arc Angle Length: Length of the arc in degrees (360 for a full ellipse)
- Arc Start Angle: Start angle of the arc in degrees (clockwise, start at right-most point)

4.6.9 Polyline and Polygon

To add or delete a point to a polyline or polygon, right-click on a point of the shape and select appropriate action on the context menu that appears.

Polygon (and Polyline) popup



Polygon Popup menu

Polygon (and Polyline) popup actions

- Add Point: Adds a point to pick-point (green-box) chosen
- Delete Point: Delete point pick-point (green-box) chosen
- Edit Point: Opens small dialog for editing point pick-point (green-box) chosen, i.e. x and y coordinate

4.6.10 Linear- and Elliptic- Tickmeter, Thermometer and Tankmeter

Data sources can be added to meters by dragging TM from the System Configuration Browser.

The meters consist of two major variations: LinearTickMeter, EllipticTickMeter and Thermometer with tick indicators, and the TankMeter with none.

The meters cannot be rotated, so placing must be done with properties on the meter elements, i.e. Linear Tick Meters gauge can be rotated, with the orientation property.

4.6.10.1 Linear Tick Meter

The Linear Tick Meter is a linear gauge with a minimum and maximum. Most of the properties of the Linear Tick Meter are appearance properties.

Linear Tick Meter properties

Appearance	
Border Color	
Border Used	☐
Label Text	
Meter Tick Indicator	Bar
Meter Tick Label Style	Left or Top
Shape Fill Color	R:0 G:0 B:0 - #000000
Shape Fill Style	None
Tick Color	R:0 G:0 B:0 - #000000
Tick Indicator Color	R:255 G:255 B:255 - #FFFFFF
Tick Indicator Thickness	6
Behaviour	
Field Style	[Disabled]
Meter Color Show Status	☑
Meter Major Tick Frequency	2
Meter Tick Base	-50
Meter Tick Unit	5
Orientation	Horizontal
Dimensions	
Font	
Field & Label Font	Lucida Sans Typewriter Regular, 0,...
Field & Label Font Color	R:0 G:0 B:0 - #000000
Tick Label Font	Lucida Sans Typewriter Regular, 0,...
Tick Label Font Color	R:0 G:0 B:0 - #000000
General	
Data Source	MeterDS1
Information	
Limits-Domain	
Meter Maximum	50
Meter Minimum	-50

Linear Tick Meter Properties in Property Editor

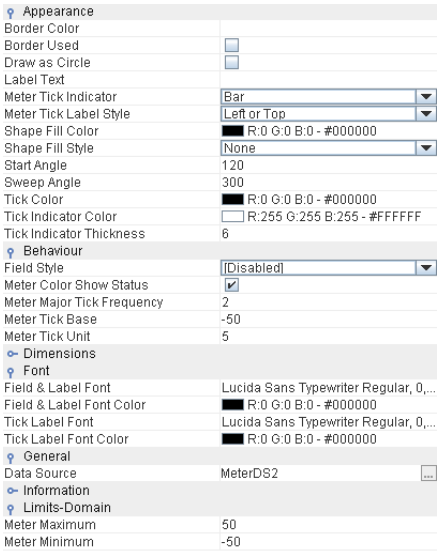
The Linear Tick Meter has several unique properties

- Border Color: Colour of border
- Border Used: If checked, border is drawn
- Label Text: Text of label shown
- Meter Tick Indicator: Type of indicator drawn
- Meter Tick Label Style: Placement of Label with respect to indicator
- Shape Fill Color: Defines the fill colour of the meter, i.e. the colour of the meter
- Shape Fill Style: Defines the fill style (background), i.e. solid = fill meter, none = no colouring of meter
- Tick Color: Color of ticks drawn
- Tick Indicator Color: Color of tick indicator drawn
- Tick Indicator Thickness: Thickness (in pixels) of tick indicator drawn
- Field Style: Location of internal data field (i.e. legend)
- Meter Color Show Status: The gauge fill colour changes to display data quality
- Meter Major Tick Frequency: Placement of major (larger) ticks
- Meter Tick Base: Placement of 'first' major (larger) tick
- Meter Tick Unit: Spacing between ticks, spacing between major ticks will be ('Meter Major Tick Frequency' x 'Meter Tick Unit')
- Orientation: Orientation of the meter/gauge
- Field & Label Font: Fonts of field and label inside meter
- Field & Label Font Color: Colors of fonts of field and label inside meter
- Tick Label Font: Font tick label inside meter
- Tick Label Font Color: Colors of font tick label inside meter
- Meter Maximum: Largest value shown in meter
- Meter Minimum: Lowest value shown in meter

4.6.10.2 Elliptic Tick Meter

The elliptic tick meter is equivalent to the Linear Tick Meter, except for one property and the appearance, i.e. drawn as an ellipse instead of a linear scale.

Elliptic Tick Meter properties



Elliptic Tick Meter properties in Property Editor

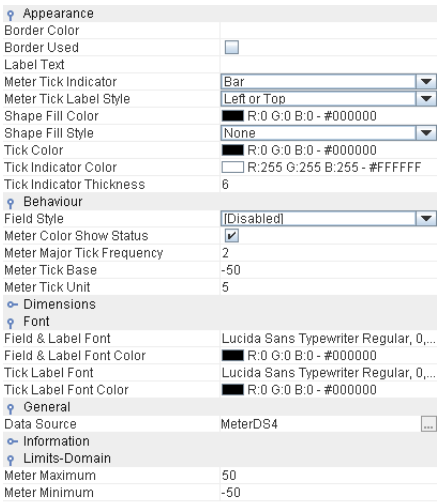
The Elliptic Tick Meter has some unique properties, but otherwise the same properties as the linear tick meter (except property: Orientation).

- Draw as Circle: Instead of drawing an ellipse, a circle is drawn instead

4.6.10.3 Thermometer

The Thermometer is equivalent to the Linear Tick Meter, except for one property and the appearance, i.e. drawn as a thermometer (kelvin and degrees Celsius drawn).

Thermometer properties



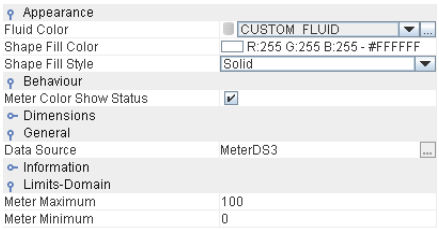
Thermometer properties in Property Editor

The Thermometer does not have any unique properties, but is a special case of the Linear Tick Meter, with the same properties (except property: Orientation).

4.6.10.4 Tank Meter

The Tank Meter element represents a fluid tank, with fluid definition.

Tank Meter properties



Tank Meter Properties in Property Editor

The Tank Meter has one unique property, otherwise the properties are the same as the linear tick meter (except fewer).

- Fluid Color: The Fluid can be predefined or a custom, i.e. color. Press mini-button: '...' to open color editing for custom fluid

4.6.11 Pipe, Valve and CheckValve

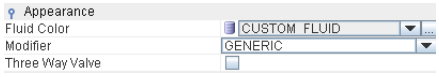
Pipes and valves can be used together to present a flow system. However, pipes and valves remain independent, unconnected elements on the drawing surface. A pipe can be positioned precisely using the cursor keys.

4.6.11.1 Valves

The Valves element represents a fluid valve.

Valve (and CheckValve) can be assigned states: INDETERMINATE, CLOSED, IN_TRANSITION, OPEN These states can be set statically and/or assign with a data source via the data source dialog. Data source assigned to the valves state property must return Strings or enumerated values, which corresponds to the states above.

Valve properties



Valve properties in Property Editor

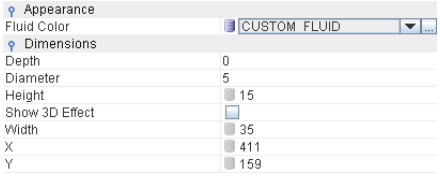
The Valves have some unique properties.

- Fluid Color: The Fluid can be predefined or a custom, i.e. color. Press mini-button: '...' to open color editing for custom fluid
- Modifier: Graphical Symbol in valve, symbolising usage
- Three Way Valve: If checked the valve is three-way instead of two-way

4.6.11.2 Pipe

The Pipe element represents a fluid pipe-system, or a group of pipe-segments.

Pipe properties



Valve Properties in Property Editor

The Pipe has some unique properties, as valves, see property: Fluid.

- Diameter: Pipe diameter, i.e. size of drawn pipes
- Show 3D Effect: Draw Pipe with 3D effect

The pipe segments can be added, deleted, or moved:

- Adding: Double-click on segment, followed by right-click on segment or end-point, selecting 'Add' from popup menu

- Deleting: Double-click on segment, followed by right-click on segment, selecting 'Delete' from popup menu
- Moving: Double-click on segment, followed a drag of segment end-points to move

4.6.12 Input Field

Input Fields can be used give an input to the system while the synoptic display is executing. Computation can refer to input fields, and use them as arguments in their computation.

Input Field properties

Input Field Properties in Property Editor

The Input Field has some unique properties, and can refer to a command in same display, acting as input to that command.

- Parameter Command: Reference to command in other element in current display
- Parameter Command Parameter: Parameter in above command

4.6.13 Input ComboBox

Like Input Fields the combobox can be used give an input to the system while the synoptic display is executing. Computation can refer to comboboxes, and use them as arguments in their computation.

Input ComboBox properties The Input ComboBox has the same properties as the input field, as well as some extra, and can refer to a command in same display, acting as input to that command.

- Inputs: The selectable inputs available in the ComboBox, possible actions are Add, Delete and Edit of seleted input

4.6.14 File Chooser

File Chooser is an extension of Input Field. It can be used to set a file name as command parameter to the Input Field. To select a file open a file browser by clicking the small button at the right side of the Input Field and browse to the file that name should be set as parameter.

File Chooser properties

- Absolute Path: Set this flag to determine that the absolute path and file name instead of file name only will be set as parameter.
- File Filter: Define file filter(s) that will be added to the file browser. A file filter is a pair of extension (e.g. "*.xml" or ".xml") and description to the file format (e.g. "XML Files (*.xml)"). The description will be added to the "Files of Type:" combo box in the file browser.
- Start Directory: Start Directory String is used to set a path to directory as starting point for the file browser. The path may contain one or more environment variables. **Note:** The string is a platform dependent path.

4.6.15 Image

The use of the image element is deprecated because external images based on the image element are stored on the file system and can not be exported to the MDB.

Instead use images (Symbols) from the Symbol Library:

1. Ensure a user library exists and is open for editing as described in the section Symbol Library.
2. On the Symbol Library click **Edit** and select **Create New Symbol**.
3. In the dialog that appears click **Add External Icon**, browse to and select the image file and click **Choose Images**.
4. Select the image in the **State Icon** drop down list.

5. Enter the name of the image in the **Symbol Name** field.
6. Click **Ok**
7. Drag the image from the Symbol Library to the Display.

Image properties

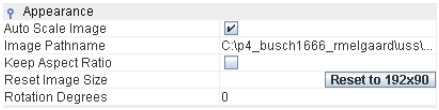


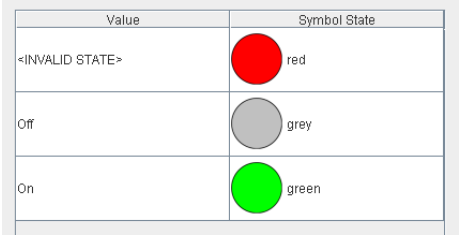
Image Properties in Property Editor

The Image element has several unique properties.

- Auto Scale Image: If checked, upon resizing the image is automatically scaled
- Image Pathname: Location of image, i.e. image source
- Keep Aspect Ratio: If checked, scaling keeps the original aspect of image, i.e. ratio between width and height of original image
- Reset Image Size: Click mini-button with original size as label to resize the image to that size
- Rotation Degrees: Rotation of image in degrees, 360 degrees for a full circle, direction clockwise

4.6.15.1 Symbol State Mapping

Special Symbol State Mapping is possible between a Symbols attached data source and the Symbol states, if the data source is a enumeration, via the Symbol State Mapper dialog:



Symbol State Mapping Dialog (available from Property Editor)

Here data source enumeration on the left can be mapped to symbol states on the right.

4.7 Data Sources

A *Data Source* bound to an element provides data to the element when the display is executed. The Editor supports binding properties of an element to a data source. When the data source provides the *value* that the element presents (e.g. the value of a Data Field or the temperature value of a thermometer), the Data Source is a *Value* property; the other possibility is a *dynamic property* which determines other properties of the element than its value (e.g. its colour or position on the display).

Another aspect of a data source is whether it is *external* or a *Computation*. External data sources are simply MDB end-items without further processing. Computations are defined by specification of an expression and can perform calculations on other data sources as described in Section 4.7.4.

4.7.1 Data Source Dialog

Data sources can be bound to a property in two different ways:

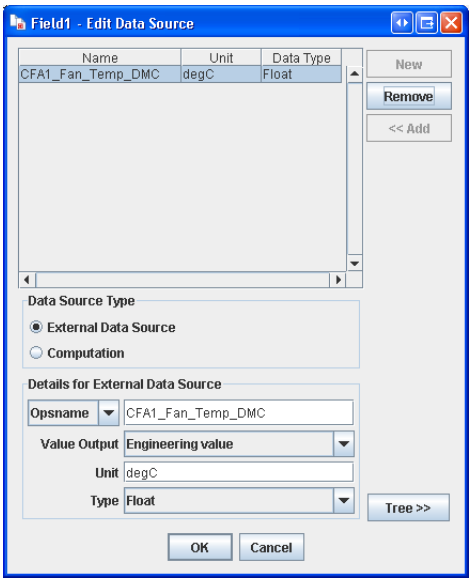
- Using drag-and-drop from the System configuration browser.
- Using the Data Source dialog as described in this section.

The Data Source dialog is opened from the Property editor by clicking the database icon (for dynamic properties) or the button with three dots (for value properties) next to the property value.

The top part of the dialog is a table of already defined data sources for the specific property of the specific element. Selecting a line in the table, the details of the selected data source are shown. The Add

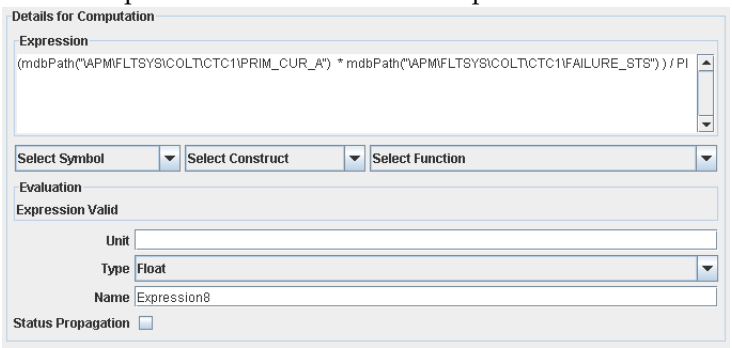
and Remove buttons are for adding and removing data sources. The Add button will be disabled when the maximum number of data sources for the property has been reached - this maximum is typically 1 - graphs accept more.

When the data source's type has been set, the lower part of the dialog will change to present the details of the selected type. External data sources are most conveniently selected using the '...' button that opens a System configuration browser. This will automatically fill in the selected end-item's name, unit and type.



Data source dialog showing details for an external data source

A computation is defined by an expression. Expressions can also contain MDB end-items and names of data sources defined in other expressions. The name of a computation cannot contain spaces.



Data source dialog details for a computation

Note that changing the type from e.g. external to computation and back to external again will clear all properties of the defined data source.

Fields

- **Unit**: Used to describe the output of your expression. Example: degC
- **Type**: The type your expression returns. Example: Integer
- **Name**: The name that describes your expression.
- **Status Propagation**: If checked, your expression will propagate statuses received from its own data sources.

4.7.2 Supported Data Source Types

The table below summarizes which data sources are supported for which elements and element properties.

Table 4.7: Supported Data Source Types and Ranges per Property and Element

Properties	Property type	Elements	Data types supported	Range / allowed values
X Y Width Height	Dynamic Property	Label Data Field Shapes Input Field Pipe	Integer Float	Non-negative
Fill Color Draw Color Corner Color	Dynamic Property	Misc.	String Enum Any	A proper formatted color-string[2]
Value	Data Source	Meters Data Field	Integer Float State-Code Any	Any range
Value	Data Source	LineGraph StripGraph	Integer Float State-Code Any	Any range
Value	Data Source	BarGraph	Integer Float Any	Any range
Value	Data Source	InputField	Any	Any range
Value	Data Source	Symbol	Any (the value after conversion to string is used)	Matching the symbol's state names[3]
Value	Data Source	Valve CheckValve	Any (the value after conversion to string is used)	"INDETERMINATE" "CLOSED" "IN_TRANSITION" "OPEN"

4.7.3 Dynamic Properties

Dynamic properties are properties whose value is controlled by a data source. The dynamic properties are listed in the section Supported Data Source Types above

The screen shot below shows the icon for dynamic properties. If the property is bound to a data source the icon becomes bright whereas unbound properties have a grey icon.

Example: If the Height property is bound to a data source that varies, the element will change its height at run-time.

Dynamic properties - the height property is bound to a data source

4.7.4 Expressions

Expressions in the USS Editor support JFormula expressions. JFormula is a library for evaluating mathematical expressions. Mathematical expressions accept the following set of operators: +, -, *, /, %, ^ where '%' stands for the modulo operator and '^' for the power operator. Parenthesis to any level is supported. The expression may contain variables as well as functions.

[2] Examples: "red", "green", "#f4a460". See other possible colors in section rgb colors.

[3] State names for predefined dynamic symbols in the USS library can be inspected as described in the section Pre-Defined Dynamic Symbols.

4.7.4.1 Operators

Supported operators for expressions are divided into several different types:

- numerical
- boolean
- string
- list
- conditional
- and other operators.

The following table explains the different operator types.

Table 4.8: Operators

Type	Operator	Example
Numerical operators	+ - * / : Basic operators % : Modulo operators ^ : Power operators	(-1 + 50*2) / (2^4)
Boolean operators	~, xor : operators && and : And operators ||, or : Or operators !, not : Not operators < : less operator > : great operator <= : less or equal operator >= : great or equal operator ==, equals : equal operators !=, <> : not equal operators	!(A && (B < 10)) | NOT (A XOR (B equals C)) A != 2 || B > 2 "string1" == "string2" A or B A or (B <> C)
String operators	== : 2 strings are equal != : 2 strings are not equal <> : 2 strings are not equal < : The first string less lexically than the second one > : The first string great lexically than the second one <= :The first string less or equals lexically than the second one >= : The first string great or equals lexically than the second one + : Concat string	"string1" == "string2" : false "string1" + "a" : "string1a" "abc" > "aaa" : true "zyx" < "bcd" : false
List operators	+ : Concat two lists - : Substract a list to another one in : Test if an element is inside a list	(1,2)+(3,4) = (1,2,3,4) (1,2) + 3 = (1,2,3) 3+(1,2)=(1,2,3) (1,2,3,4)-(3,4)=(1,2) (1,2,3,4)-3=(1,2,4) 2 in (1,2,3)=true 4 in (1,2,3)=false
Other operators	= : set a variable operator [] : absolute value ^{2} : power 2 operator % : Percent operators	A = [2 - A] * 2 2^2 10%=0.1
Conditional operators	if then if then else	if (A > 2) then "Ok" if (A <=2) THEN B=3 else B=4

4.7.4.2 USS Library Extensions

mdbPath: Returns the engineering value for the data source with the given Pathname:
(Note that the pathname argument has to be specified in quotes. See sample expressions.)

```
Variant mdbPath( String pathname )
{
  engineeringValue = getDataFor(pathname).getValue();
  return( engineeringValue )
}
```

transformStateCode: Transforms a Columbus MDB state code into an IDAGS conformant state code:
(IDAGS conformant state codes consist of one or multiple literals separated by space. All literals begin with a capital letter.)

```
String transformStateCode( String stateCode )
```

4.7.4.3 Functions

It is possible to execute various functions in an expression. Functions can be selected from the dropdown menu "Select Function". The following functions include operations on byte buffers. Bit positions start with value 0 and increase from left to right. For example, bit position 0 denotes the leftmost bit in the first byte of a buffer, position 7 is the rightmost bit in the same byte. Position 8 is the leftmost bit in the second byte of the buffer and so on. When multiple byte are interpreted together then big endian (or network) byte order is assumed that is the most significant byte (MSB) value is stored at the memory location with the lowest address in the buffer.

BufferGetBit: Returns the Double value 1 if the bit in the byte buffer at the specified position is set. Otherwise the Double value 0 will be returned.

Definition:

```
bufferGetBit(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetBit("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> 1
```

BufferGetSigned32: Returns the Double value 1 if the bit in the byte buffer at the specified position is set. Otherwise the Double value 0 will be returned.

Definition:

```
bufferGetSigned32(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetSigned32("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> 2
```

BufferGetTime: Returns UTC time created from a 40 bit GPS time from the byte buffer at the specified bit position as formatted String ("ddMMMyy HH:mm:ss"). GPS time is extracted from the byte buffer as 40 bit struct. coarse time : bit 0 .. 31 int (seconds since 06.01.1980) fine time : bit 32 .. 40 ubyte (1 / 256 seconds).

Definition:

```
bufferGetTime(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetTime("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8)
```

BufferGetUnsigned16: Returns the unsigned short value from the byte buffer at the specified bit position as double value.

Definition:

```
bufferGetUnsigned16(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetUnsigned16("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> 2
```

BufferGetUnsigned32: Returns the unsigned int value from the byte buffer at the specified bit position as double value.
Definition:

```
bufferGetUnsigned32(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetUnsigned32("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> 2
```

BufferGetUnsigned8: Returns the unsigned byte value from the byte buffer at the specified bit position as double value.
Definition:

```
bufferGetUnsigned8(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferGetUnsigned8("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> 2
```

BufferIsBitSet: Returns the boolean value for the bit in the byte buffer at the specified position.
Definition:

```
bufferIsBitSet(parameter : String, bitPosition : Integer) : Integer
```

Example:

```
bufferIsBitSet("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) -> true
```

Checkmark: Returns a checkmark character when the given input string is equal to the compare String argument. Otherwise and empty String is returned.
Definition:

```
checkmark(input : String, compare : String) : String
```

Example:

```
checkmark("PUMP_ON", "PUMP_ON") -> "v"
```

MdbPath: Computes the engineering value of a parameter specified by an MDB path (as string constant). NOTE: This function is deprecated and provided only for compatibility with previous uss versions. Use parameterEngineeringValue(path : String) instead.
Definition:

```
mdbPath(path : String) : Object
```

Example:

```
mdbPath("\APM\FOO\ITEM") -> 3.0
```

ParameterAcquisitionStatus: Returns the acquisition status of a parameter specified by a string constant.
Definition:

```
parameterAcquisitionStatus(parameter : String) : String
```

Example:

```
parameterAcquisitionStatus("CTCU1_Cabin_Temp1_DMC") -> "ACQUIRED"
```

ParameterCautionViolationThreshold: Returns the limit caution violation threshold of a parameter specified by a string constant.
Definition:

```
parameterCautionViolationThreshold(parameter : String) : Integer
```

Example:

```
parameterCautionViolationThreshold("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterDeltaMonitoringStatus: Returns the delta monitoring status of a parameter specified by a string constant.
Definition:

```
parameterDeltaMonitoringStatus(parameter : String) : String
```

Example:

```
parameterDeltaMonitoringStatus("CTCU1_Cabin_Temp1_DMC") -> "IN_LIMITS"
```

ParameterExpectedValue: Returns the expected value of a parameter specified by a string constant.
Definition:

```
parameterExpectedValue(parameter : String) : String
```

Example:

```
parameterExpectedValue("CTCU1_Cabin_Temp1_DMC") -> "foo"
```

ParameterExpectedValueViolationThreshold: Returns the limit caution violation threshold of a parameter specified by a string constant.
Definition:

```
parameterExpectedValueViolationThreshold(parameter : String) : Integer
```

Example:

```
parameterExpectedValueViolationThreshold("CTCU1_Cabin_Temp1_DMC") -> 2
```

ParameterHighCautionLimit: Returns the high caution limit of a parameter specified by a string constant.
Definition:

```
parameterHighCautionLimit(parameter : String) : Number
```

Example:

```
parameterHighCautionLimit("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterHighWarningLimit: Returns the high warning limit of a parameter specified by a string constant.
Definition:

```
parameterHighWarningLimit(parameter : String) : Number
```

Example:

```
parameterHighWarningLimit("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterLimitId: Returns the limit identifier of a parameter specified by a string constant.
Definition:

```
parameterLimitId(parameter : String) : Integer
```

Example:

```
parameterLimitId("CTCU1_Cabin_Temp1_DMC") -> 2
```

ParameterLowCautionLimit: Returns the low caution limit of a parameter specified by a string constant.
Definition:

```
parameterLowCautionLimit(parameter : String) : Number
```

Example:

```
parameterLowCautionLimit("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterLowWarningLimit: Returns the low warning limit of a parameter specified by a string constant.
Definition:

```
parameterLowWarningLimit(parameter : String) : Number
```

Example:

```
parameterLowWarningLimit("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterMonitoringStatus: Returns the monitoring status of a parameter specified by a string constant.
Definition:

```
parameterMonitoringStatus(parameter : String) : String
```

Example:

```
parameterMonitoringStatus("CTCU1_Cabin_Temp1_DMC") -> "IN_LIMITS"
```

ParameterProcessingStatus: Returns the processing status of a parameter specified by a string constant.
Definition:

```
parameterProcessingStatus(parameter : String) : String
```

Example:

```
parameterProcessingStatus("CTCU1_Cabin_Temp1_DMC") -> "PROCESSED"
```

ParameterRawValue: Returns the raw value of a parameter specified by a string constant.
Definition:

```
parameterRawValue(parameter : String) : Object
```

Example:

```
parameterRawValue("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

ParameterValue: Returns the engineering value of a parameter specified by a string constant.
Definition:

```
parameterValue(parameter : String) : Object
```

Example:

```
parameterValue("CTCU1_Cabin_Temp1_DMC") -> 20.0
```

ParameterWarningViolationThreshold: Returns the limit warning violation threshold of a parameter specified by a string constant.
Definition:

```
parameterWarningViolationThreshold(parameter : String) : Integer
```

Example:

```
parameterWarningViolationThreshold("CTCU1_Cabin_Temp1_DMC") -> 2.0
```

TransformStateCode: Transforms a string into an OpNom conform string. Words will be properly capitalized, and underscores will be replaced with blanks.
Definition:

```
transformStateCode(s : String) : String
```

Example:

```
transformStateCode("PUMP_ON") -> "Pump On"
```

4.7.4.4 Selected Computations from PREP Library

The following computations from the PREP library are available:

a1EqualToInt: Returns a value of true if the value from the PUI is equal to the constant int value. Pseudo code:

```
int alEqualToIntComp( int valve, int valve_state )
{
  If (valve_state = valve)
    result =  1
  else
    result =  0
  return( result )
}
```

ANDComp: Takes the input arguments and Logically ANDs them together. Pseudo code:

```
int ANDComp (int arg1, int arg2, int arg3 .. int argN)
{
  result = (arg0 and arg1 and arg2 and and arg(N-2) and arg(N-1) and argN)
  return( result )
}
```

GreaterThanConstComp: Determine if value of input PUI is greater than the supplied constant value. Pseudo code:

```
boolean GreaterThanConstComp (float PUI, const float CONSTANT)
{
  if(PUI > CONSTANT) return true;
  else return false;
}
```

MultiCompareComp: Does multiple comparisons. If all comparisons are true, return true. Otherwise return false. Every PUI should be associated with a CONST. Pseudo code:

```
boolean sdsPipeComp(int PUI1, int CONST1, int PUI2, int CONST2,
                    ... int PUIn, int CONSTn)
{
  // return true if and only if all comparisons are true
  if ((PUI1 == CONST1) AND (PUI2 == CONST2) AND ... AND (PUIn == CONSTn)
    return = 1
  else
    return = 0
}
```

OrComp: Takes the input arguments and calculates their sum.

```
int ORComp( arg0, arg1, ..., arg(N-1), argN)
{
    result = (arg0 or arg1 or arg2 or ... or arg(N-2) or arg(N-1) or argN)
    return( result )
}
```

4.7.4.5 Examples

In this chapter we introduce some typical USS expression examples.

Example 4.1 Using the engineering value in expressions

The following examples show how engineering values are accessed in an expression. In this example a data source is referenced via its Opsname.

```
IF (CTCU1_Cabin_Temp1_DMC >= 20) THEN "Warm" ELSE "Cold"
```

The same expression using the path name instead of the Opsname (Note that the path name must be surrounded with quotes).

```
IF (mdbPath("\APM\FLTSYS\ECLSS\CTCU1\CABIN_TEMP1") >= 20) THEN "Warm" ELSE "Cold"
```

Example 4.2 Group-of-bytes

This example shows how to retrieve the flap execution state value of type unsigned8 from a flap status SW variable. The value starts at bit position 8 in an otherwise untyped byte buffer.

```
if bufferGetUnsigned8("USM_DMC_UCL_Slot1_FLAP_Stat_SW", 8) == 2 then "Running"
```

Example 4.3 Using expressions to set colors

This example shows how to set the background color of a rectangle depending on a parameter value. The result type of the expression is a string. The string value must evaluate to a valid color specifier. Valid colors can be specified as RGB values in hexadecimal format with a hash sign as prefix or as color names.

```
if ds_SIM == "RT_NORM" then "#86B78A" else "#B88687"
```

Here is the same example with explicit color names. See Section 9.4 for the full list of color names.

```
if ds_SIM == "RT_NORM" then "azure" else "plum"
```

Example 4.4 Mapping integers to strings

This is a NASA PCS Example, which converts an integer parameter referenced by PUI to a string value. Note how if-expressions can be nested.

```
IF (EMDC01SW1060J == 0) THEN "On" ELSE IF (EMDC01SW1060J == 1) THEN "Off" ELSE " ↩
    ERR!"
```

Example 4.5 Constant expression

This example shows how a constant value can be defined as a data source. If this computation is connected to a data field, the field will always show the value 42.

```
42
```

Example 4.6 Multi-line expression

One expression is normally written in a single line. Multiple expressions can be used to specify more complex computations. Also, variables can be used to store intermediate results. This example has two expressions each on a separate line. The variable `a` is assigned the constant value 3. The variable `b` is then computed by adding one to the value of `a`. The result of the whole computation is the result of the last expression (here it is `b` which has the constant value 4).

```
a = 3
b = a + 1
```

4.8 Mission Database

The settings for the MDB connection are made in Preferences, use the Project root folder setting to define: Where MDB displays are stored. If displays are kept locally in different locations, the USS project folder root has to be set when changing local store point (i.e. USS only maintains one project folder root).

Symbol Libraries can be handled via the CGS MDB, see MDB Version Control of Symbol Libraries.

TIP

USS Editor can work closely or loosely with the CGS/MDB:

- closely - launched via the IMDB (displays are automatically synchronized - double click on USS_DISPLAY end-item will launch the USS Editor with display), see CGS Usermanual for more information, when USS Editor is launched from IMDB (Autoload-mode) some parameters are preset / locked to that IMDB setup:
  - MDB Settings in general, as well as CCU/CDU configuration
  - SCOE file is locked to a snapshoot of the MDB used by the IMDB setup
- loosely - USS Editor is launched locally and edits files locally (which can be saved etc. via MDB interface)

4.8.1 Opening Displays from the MDB

The USS Editor can open USS displays from the MDB, stored in a tree-like structure containing the displays. The open function covers to possible actions sync (synchronize) and lock-for-edit (locking of display in MDB)

Synchronization: When an updated version of a display is needed (latest version of display in MDB), synchronization is performed from the Editor. When invoking **File | MDB | MBD Browser (Sync/Lock-For-Edit)...**, an MDB Browser is opened. The MDB Browser shows the tree-structure of the MDB and here multiple displays can be selected by navigating the tree. The selected displays can be synchronized, by right-clicking and selecting **sync**. Click **Close** to quit the MDB Browser. Synchronization is always done to the Project Folder.

Lock-for-Edit: When a display needs to be locked for edit (locking the display in MDB, ensuring only one person edits the display at a time), lock-for-edit is performed from the Editor. In the MDB Browser the selected displays can be lock-for-edit, by right-clicking and selecting **lock-for-edit**, this will also synchronize the selected displays.

Results of Lock-for-edit action on a display:

- In MDB: The display (end-item) is locked to the database user. This results in nobody else being able to lock it and only the person, who locked the display in the first place, can submit an updated display (unlocking it in doing so).
- Locally / on the file system: An updated version display is copied to the file system (placed appropriately in project-root-folder), and the display-file is changed to read-writable for edit.

4.8.2 Saving Displays in the MDB

The Editor can save USS displays in the MDB, **Note:** Only displays locked for edit can be saved in the MDB.

4.8.3 Adding a New Display to the MDB

When adding a new display to the MDB, first ensure that the MDB configuration is correct (refer to Preferences and take care of the CU Version setting, which is important when adding displays to the MDB.

The editor will on submitting check for the following MDB naming constraints: Display names can be a maximum of 16 characters in MDB notation, file names must be in uppercase, and file names cannot contain a minus (-).

Do the following steps to add a new display to the MDB:

1. Create the new display
2. Save the display to a location in the project folder equivalent to the MDB path. Use only capital letters for the display name and no more than 16 characters, otherwise the MDB will not accept it. E.g. the file system path *ProjectFolder*\ CDU_316990119\ APM\ COMMON_TEST_SUPP\ GRD_DATA\ USS\ EDITOR\ MANUAL_TESTS is equivalent to MDB path (for CDU with Id: 316990119) \APM\ COMMON_TEST_SUPP\ GRD_DATA\ USS\EDITOR\ MANUAL_TESTS. *ProjectFolder* is configured in Preferences.
3. Right-click on the display (in Synoptic Hierarchy or on the display itself) to get a popup menu.
4. In the popup-menu choose **MDB | Add to MDB** and the Editor will then:
   - Connect to MDB. While the Editor is connecting to the MDB a "Connecting to MDB" dialog is visible, this can take up to 10 minutes depending on the connection.
   - Locate the display path in MDB
   - Add the display to the MDB
   - On success the Editor will display a confirmation, on error an error message.

4.8.4 Forced Open From MDB (revert)

Forced Synchronization: When undoing of an action is needed (reverting to latest version of display in MDB), forced synchronization is performed from the Editor. Open the previously synchronized or lock-for-edit display, located in the project folder. When invoking **File | MDB | Forced Open From MDB (revert)**, the Editor will overwrite the display with the MDB version and set display-file to read-only in the local file system. NB! The edited display-file will be overwritten and cannot be recovered after this operation (i.e. can be seen as a reopen of a changed display, reverting changes).

4.8.5 Delete in MDB

Deletion of display: When deletion of a display is needed (removing of display from MDB), delete in MDB is performed from the Editor. Open the previously synchronized display, located in the project folder. When invoking **File | MDB | Delete in MDB**, the Editor will delete the display in the MDB and delete the locally stored display-file in the local file system. NB! The display-file will be lost and cannot be recovered after this operation.

4.9 System Configuration Browser

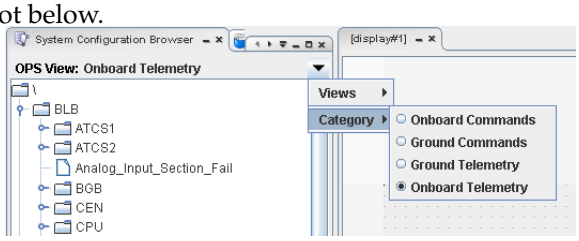
The System configuration browser (SCB) shows a tree view of the SCOE file selected on the MDB-SCOE page in Preferences. The SCB is used to drag observable (TM) and activatable end-items (TC) onto the display or elements thus creating, adding or updating data sources as described in the table below.

Table 4.9: Results of dragging TM/TC from the SCB

Target	**Result**
TM to Display area	Create Data Field (TM). Multiple TM can be dragged at once.
TC to Display area	Create Command Button (TC). Multiple TC can be dragged at once.
TM to **selected** Data Field	Update data source for the field.
TC to Command List	Add TC to the list.
TM to graph	Add TM as data source for the graph. TM must be of type state code, integer of float.
TM to meter	Update data source for the meter. TM must be of type integer of float.

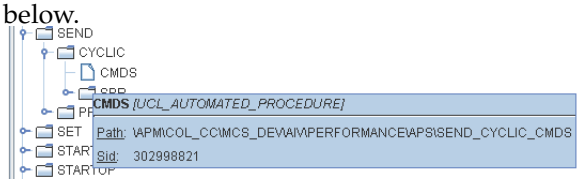
The current system configuration version (CCU internal version) is shown as a tooltip in System Configuration Browser by holding the mouse over title "PATH/OPS View: ...".

The View (Path or Ops) and the Category (onboard/ground TM/TC) is switched with the combo box above tree -- see screen shot below.



System Configuration Browser - toggling category

Detailed information about an end item can be obtained by right clicking the end item and selecting Properties. The tool tip that appears when holding the mouse over the end item gives SID and Path information -- see screen shot below.



System Configuration Browser context menu

The SCB contains a search function at the bottom: enter the string to search for and click the **Search** button. To search again press F3.

4.10 Working with Projects

4.10.1 Synoptic Hierarchy

The Synoptic Hierarchy is a tool-view for the Editor to ease navigation of currently open displays and the USS project displays (defined by the project root-folder). It provides possibility to create navigation target to project displays, and easy opening hereof. Furthermore it provides rename, delete and explorer functionality

The Synoptic hierarchy displays two structures:

- Open Displays: The displays currently open in the Editor. Below each display all elements on the display are shown. This view can also be used to drag elements between displays and to inspect compound objects for their contents.
- Synoptic Hierarchy: the file folder structure for the XML based USS displays files below the Project root folder. This view can be used to easily create navigation buttons by right click a display in the hierarchy and selecting **Add Navigation to display**.

Displays in the upper list are already open. Click a display in this list to set focus to it. The displays in the second list can be opened by right clicking and selecting **Open Display**.

4.10.2 Consistency Check

By selecting the Consistency check from the Tools menu, the current display is checked against the current SCOE file. A report is generated listing all inconsistencies. Standard checks are:

- For all data sources Opsname, SID and Pathname are checked if they match each other. If the Opsname is found but the SID or Pathname does not match the data in the SCOE file then the SID or Pathname found in the SCOE file is proposed in the report.
- The data source Unit is checked if it matches the data in the SCOE file.
- The data source Type is checked. If the data type is enum the data is checked if it contains all the statecodes in the right order.
- Computation-data sources will be checked if they return any syntax errors.

- Displays referenced as navigation target by elements (e.g. navigation buttons) at first is checked if the display exists in file system. Second it is checked how navigation target is defined. In case it is defined as 'Filename' consistency check is finished. In other case it is defined as 'Alias' (e.g. OPS NAME) or 'MDB Path' it is checked if type of display is 'USS DISPLAY', then is checked if 'Alias' or 'MDB Path' exist in SCOE files and at least if the path in SCOE file still maps the path in file system.
- A labels that is associated with a field-data source and is set to show opsname or pathname will be checked that the label text shows that opsname or pathname.
- Opnom check. Checks that all label texts conform to the OpNom definition or English as set in the Preferences.
- It is checked that all element names are unique.

Using the LAPAP Profile additionally performs another set of consistency checks:

- For MDB end items verify that all identifiers (OPS name, pathname, SID) are defined, except for command parameters which are identified via OPS name only.
- For PWS synoptic displays only structured commands are allowed.
- Check of commands w.r.t. parameter constraints.
- Check if mandatory parameters of commands are available.
- Check if order of command parameters is inline with MDB definition.
- Symbol libraries and symbols are readable.
- Check that the synoptic display has OPS name and display title defined.
- Configurable target system check of the synoptic displays (for LAPAP MK2 only PWS allowed).

Details about consistency results, the messages, and actions to take in case of errors:

Table 4.10: USS display consistency checker messages

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for Display Target: Target System of display: XXX mismatch with expected: PWS	The only display target system allowed in LAPAP context is 'PWS'.	In display editor update display property 'Target System'.
LAPAP	ERROR	ERROR for Display Title: Display has no title	In LAPAP context the display title is mandatory.	In display editor update display property 'Title'.
LAPAP	ERROR	ERROR for Display OPS Name: Display has no Ops name	In LAPAP context the display opsname is mandatory.	In display editor update display property 'Opsname'.
STANDARD / LAPAP	ERROR	ERROR for <Field> XXX: Invalid computation expression: YYY	The data source of field XXX is a computation. Its computation expression 'YYY' is invalid.	In display editor part 'Edit data source' enter correct expression. The expression is validated simultanously.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource Opsname not defined.	Mandatory identifier opsname is missing.	In display editor part 'Edit data source' enter data source opsname.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource Pathname not defined.	Mandatory identifier pathname is missing.	In display editor part 'Edit data source' enter data source pathname.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource Opsname not in SCOE data.	The opsname identifier of the data source can not be found in the SCOE data files selected as context.	In display editor part 'Edit data source' enter correct data source opsname. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource Pathname not in SCOE data.	The pathname identifier of the data source can not be found in the SCOE data files selected as context.	In display editor part 'Edit data source' enter correct data source pathname. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource SID not defined.	Mandatory SID is missing.	In display editor part 'Edit data source' enter data source SID.
LAPAP	ERROR	ERROR for <Field> XXX: Data source: DataSource SID not in SCOE data.	The SID identifier of the data source can not be found in the SCOE data files selected as context.	In display editor part 'Edit data source' enter correct data source SID. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for <Field> XXX: Data source: SID, opsname, and path are inconsistent to SCOE data.	The three identifieres SID, opsname, and pathname, are pointing to an item in the SCOE data where one or two of them are different.	In display editor part 'Edit data source' enter correct data source identifiers. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.
STANDARD / LAPAP	ERROR	ERROR for <Field> XXX: Data source: Wrong Type: 'YYY' does not match 'ZZZ' (CGS type is 'TTT').	Data type of field (YYY) does not match data type of data source (ZZZ). CGS data type (TTT) is given for information.	In display editor part 'Edit data source' enter correct data type. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.
STANDARD / LAPAP	ERROR	ERROR for <Field> XXX: Data source: Wrong Type: 'YYY' does not match 'Unknown data type' (CGS type is 'TTT').	Data type of field (YYY) does not match data type of data source (unknown). CGS data type (TTT) is given for information.	In display editor part 'Edit data source' enter correct data type. OR: In display editor select Edit --> Preferences --> SCOE and set appropriate SCOE version.
STANDARD / LAPAP	ERROR	ERROR for <Field> XXX: Data source: Unit missing should be 'YYY'.	Engineering unit of data source is missing. Correct unit from SCOE data is YYY.	In display editor part 'Edit data source' enter correct engineering unit. OR: In display editor select edit --> Preferences --> SCOE and set appropriate SCOE version.
STANDARD / LAPAP	ERROR	ERROR for <Field> XXX: Data source: Wrong Unit: 'YYY' should be 'ZZZ'.	Engineering unit of data source (YYY) differs from engineering unit found in SCOE data (ZZZ).	In display editor part 'Edit data source' enter correct engineering unit. OR: In display editor select edit --> Preferences --> SCOE and set appropriate SCOE version.
LAPAP	WARNING	WARNING for <Field> XXX:	Field has no data source attached.	Update field in display editor and add data source definition.
LAPAP	INFO	for Info: Validation of nnn data-sources passed (mmm datasources are Computations)	Summary of validation of data sources of one display. No invalid data found.	
LAPAP	INFO	for Info: Validation of nnn data-sources FAILED with mmm errors (kkk datasources are Computations)	Summary of validation of data sources of one display. Number of errors found = mmm.	

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand is not of type 'structured'.	In LAPAP context only structured telecommands are allowed.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand has invalid command structure: YYY	Structured command does not comply to FLAP/SWOP structure.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand of type FLAP uses invalid command call: YYY	Only valid command call of FLAP is 'execute_flap'.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand of type SWOP uses invalid command call: YYY	Only valid command call of SWOP is 'issue_sw_command'.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter must be an opsname identifier: YYY	Parameter YYY is of type 'opsname identifier' instead of type being assigned in the command definition.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Invalid opsname identifier for SYSTEM_CMD_CHECK: YYY	Parameter SYSTEM_CMD_CHECK was given an invalid value YYY.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter name ONBOARD_NODE invalid for command type YYY	Parameter ONBOARD_NODE is only allowed for telecommands of type SWOP.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter name ONBOARD_EXECUTION_NODE invalid for command type YYY	Parameter ONBOARD_EXECUTION_NODE is only allowed for telecommands of type FLAP.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter name invalid: YYY	Parameter YYY is not a valid parameter for FLAP or SWOP commands.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter value of type identifier missing: YYY	Parameter YYY has no value assigned. Value must be an opsname identifier.	In display editor part 'Edit Telecommand' enter correct parameter opsname. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand: Opsname does not reference an existing item: YYY	The opsname YYY does not reference a telecommand item in the SCOE data.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand: Display command type not matching MDB command type.	UCL_AUTOMATED_PROCEDURE item must be attached to FLAP structured command. SWOP_COMMAND item must be attached to SW_CMD structured command.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand: Invalid onboard command type.	Command type is not a valid onboard command in LAPAP context.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: TeleCommand: Opsname does not reference an onboard command item: YYY	Opsname YYY does not reference a FLAP or SWOP command item.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Value for mandatory parameter missing: YYY	Mandatory command parameter YYY is not given a value in the command definition.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Invalid parameter sequence: XXX	Command parameters are set in the wrong order compared to the definition in SCOE data files.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Not a valid parameter: YYY	Parameter YYY does not match with any parameter in the command definition of SCOE data.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Parameter constraint violation: YYY	Parameter YYY has an invalid value that violates the predefined constraints of it as to be found in the SCOE data files. Constraints could be a range or a set of given values.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
LAPAP	ERROR	ERROR for <CommandButton> XXX: Type of parameter does not match parameter definition in MDB: YYY	Parameter YYY is of invalid type compared to the type definition in the SCOE data files.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	ERROR	ERROR for <CommandButton> XXX: Display parameter constraint violates SCOE definiton of constraint: YYY	Parameter YYY overwrites the predefined constraint definition of the SCOE data files. That new constraint violates the limits of the original constraint.	In display editor part 'Edit Telecommand' enter correct structured command. OR: In display editor select valid onboard command from System Configuration Browser.
LAPAP	WARNING	WARNING for <CommandList> cmd: XXX: TeleCommand undefined	Telecommand definition is missing for label 'XXX'.	Create new Telecommand in display editor.
LAPAP	INFO	for Info: Validation of nnn telecommands passed	Summary of validation of telecommands of one display. No invalid data found.	
LAPAP	INFO	for Info: Validation of nnn telecommands FAILED with mmm errors	Summary of validation of telecommands of one display. Number of errors found = mmm.	
STANDARD / LAPAP	REPORT	for Info: Starting consistency check on: DISPLAY_TITLE (File: DISPLAY_FILENAME)	Summary of validation of telecommands of one display. Number of errors found = mmm.	
STANDARD / LAPAP	REPORT	for Info: Detailed Version: SYSTEM_CONFIGURATION_VERSION	Reports the SCOE version used in the consistency check.	
STANDARD / LAPAP	REPORT	for Info: SCOE files used: SCOE_FILES	Reports the SCOE files used in the consistency check.	
STANDARD / LAPAP	INFO	for Info: OpNom check disabled	Inform that OpNom spell check is disabled.	
STANDARD / LAPAP	INFO	for Info: OpNom Spell Check Performed: XXX	Inform if an OpNom spell check have been performed. Spelling errors will be listed but not classified as consistency errors.	
STANDARD / LAPAP	ERROR	The element name 'XXX' is used by more than one element.	Element names must be unique.	Rename one of the elements by editing the Name property in the Property Editor.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
STANDARD / LAPAP	WARNING	Dynamic property of color type (e.g. fill color) not bound to enum or string		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	WARNING	Dynamic property of integer type (e.g. X, Y, width, heigth) bound to a non integer or float datasource		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	WARNING	Dynamic symbol expecting an enum and getting int or float measurement		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	ERROR	Library 'XXX' are missing		Make sure that the USS editor project root folder is correct. Edit -> preferences, Project - Root folder.
STANDARD / LAPAP	ERROR	Symbol 'XXX' in library 'YYY' are missing		Open Symbol Library, select library YYY and recreate Symbol XXX.
STANDARD / LAPAP	ERROR	Image(s) XXX in symbol 'YYY' in library 'ZZZ' are missing		Open the Symbol Library, select the symbol and right click Edit Symbol. Here you can try to fix the symbol by adding the missing picture.
STANDARD / LAPAP	WARNING	Meter expecting a float or integer value but the assigned dataource is of type enum or string		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	WARNING	Valve or Check Valve expecting an enum and getting int or float measurement		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	WARNING	Pipe expecting an enum and getting int or float measurement		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	ERROR	LineGraph containing enum and int or float measurement together		Correct data type in the Data Source property in the Property Editor
STANDARD / LAPAP	WARNING	Label text inconsistent with context		Double click on label to change text.
STANDARD / LAPAP	WARNING	Label text inconsistent with basename		Double click on label to change text.
STANDARD	ERROR	Unknown symbol : XXX	The data source of field is a computation. Its computation expression 'XXX' is invalid.	In display editor part 'Edit data source' enter correct expression. The expression is validated simultanously.

Table 4.10: (continued)

Check Profile	Severity Level	Error Message	Comment	How to Fix
STANDARD	ERROR	Unknown function : XXX	The data source of field is a computation. Its computation expression 'XXX' is invalid.	In display editor part 'Edit data source' enter correct expression. The expression is validated simultanously.
STANDARD	ERROR	The Opsname XXX is not known in the current configuration.	Mandatory identifier opsname is missing.	In display editor part 'Edit data source' correct data source opsname.
STANDARD	ERROR	The SID XXX is not known in the current configuration.	Mandatory identifier SID is missing.	In display editor part 'Edit data source' correct data source SID.
STANDARD	ERROR	The Pathname XXX is not known in the current configuration.	Mandatory identifier pathname is missing.	In display editor part 'Edit data source' correct data source pathname.
STANDARD	ERROR	Opsname, SID or Pathname must be defined.	Mandatory identifier Opsname, SID or Pathname is missing.	In display editor part 'Edit data source' enter a data source.
STANDARD	ERROR	Data source: Inconsistent naming, none of opsname, pathname or SID defined for datasource.	Mandatory identifier Opsname, SID or Pathname is wrong.	In display editor part 'Edit data source' correct data source.
STANDARD	ERROR	Data source: Error: XXX for YYY	An internal read exception occured, XXX will give a description.	Inspect the error reported in XXX
STANDARD	WARNING	Parameters checked with warnings: Inconsistent naming, pathname: Pathname should not be defined but is set to: 'XXX'. for datasource with Opsname "YYY"		Correct Data Source in Property Editor.
STANDARD	WARNING	Parameters checked with warnings: Inconsistent naming, pathname: 'XXX' should be 'YYY'. for datasource with Opsname "ZZZ"		Correct Data Source in Property Editor.
STANDARD	OPNOM-DEVIATION	Parameters checked with opnom deviations: Inconsistent naming, opsname is empty. for datasource with Opsname "XXX"		Correct Data Source in Property Editor.
STANDARD	OPNOM-DEVIATION	Parameters checked with opnom deviations: Inconsistent naming, opsname: Opsname should not be defined but is set to opsname: 'XXX'. for datasource with Opsname "YYY"		Correct Data Source in Property Editor.

Table 4.10: (continued)

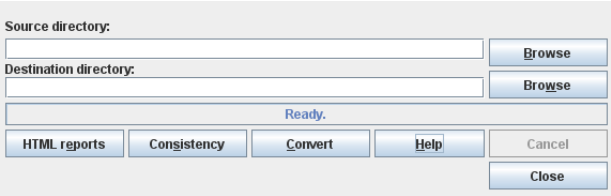
Check Profile	Severity Level	Error Message	Comment	How to Fix
STANDARD	OPNOM-DEVIATION	Parameters checked with opnom deviations: Inconsistent naming, opsname: 'XXX' does not match pathname: 'YYY'. for datasource with Opsname "ZZZ"		Correct Data Source in Property Editor.
STANDARD	OPNOM-DEVIATION	Parameters checked with opnom deviations: Inconsistent naming, opsname: Opsname should not be defined but is set to opsname: 'XXX'. for datasource with Opsname "YYY"		Correct Data Source in Property Editor.
STANDARD / LAPAP	ERROR	Navigation target is not defined.		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	Navigation target 'XXX' does not exist. On this machine the full name of this display would be: 'YYY'		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	Alias 'XXX' is not defined for navigation target 'YYY'.		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	Alias 'XXX' hasn't required type USS_DISPLAY but YYY.		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	Display 'XXX' doesn't match referenced path 'YYY' of alias 'ZZZ'.		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	MDB path 'XXX' not defined for navigation target 'YYY' .		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	MDB path 'XXX' hasn't required type USS_DISPLAY but YYY.		Correct Target in Property Editor.
STANDARD / LAPAP	ERROR	Bad field format "XXX" (YYY)	Formating parameters in Property Field Data Formatting in the Property Editor are wrong.	Refer to the section on Field Data Formatting in the usermanual.

4.10.3 Batch operations

By selecting 'Run Batch Operation ...' from the Tools menu, the editor can be used to make the same operation on multiple displays.

Batch Operations include:

- Generation of HTML reports
- Consistency Check
- Convertion to USS display format



Batch Operation Dialog

How-to use the Batch Operation Dialog:

1. Write the source and destination directories in the two input fields (or use the browse button)
2. Click on one of the batch execution buttons 'Convert', 'Consistency' or 'HTML reports'

TIP

Please note the following when using the batch operation dialog:

- Please note that source and destination directories must exist
- Errors occuring during the execution must currently be examined in the console or the log
- If the output files already exist they will be replaced

4.10.4 Auto Saver

The editor has a auto-save function, which stores save-displays being edited to the location they are save. The filename of the auto-saved display is concatenated with a tilde (~), to distinguish auto-saved displays from the original ones.

EX: '/home/test/uss-project/test.uss' will be auto-saved to '/home/test/uss-project/test.uss~'

By default the editor will auto-save every 300 seconds, see Preferences for configurating the auto-save function.

TIP

Please note the following:

- Since the auto-saver saves only saved displays, a new display should be saved, when constructed to make it part of the auto-saved displays
- If errors should occure, while unsaved changes exist, choose to continue the application, and save changes to a new display file-name (not over-writting the old)
- Opening and editing an extensive amount of displays (more than 20), might cause the auto-save function to not meet it fail-save max-execution-time. To minimize this risk save current display, before changing to editing other display
- Max-execution-time can be set manually in uss.properties. Ex: for maximum execution time to 10 seconds add a line with: uss.editor.autosave.maxexecutiontime=10000

4.11 Configuring the Editor

The Editor can be configured in two levels: The system settings with are set by the administrator of the tool and the User Preferences set by the display author.

4.11.1 System Setting

System settings are defined in the uss.properties file which is common for the Editor and Executor. The uss.properties file can be edited in any standard text editor.

4.11.2 Preferences

The preferences setting are accessed though **Edit | Preferences...** on the menu. These are the preference pages:

- **Project**: Project settings for editor. Contains project root settings, and settings for auto save function, see the Auto Saver.
  - **Project**: The root folder for the synoptic hierarchy
  - **Autosave Interval**: Determines the time the auto-save function shall wait, before re-auto-saving open project displays (default 300 seconds).
  - **Autosave Enabled**: Determines if the auto-save function for the editor shall run (default on).
- **Display**: Grid and zoom settings for new displays. To change the settings for the current display, use the View Settings.
- **Consistency**: OpNom setting for the Consistency Checker. The consistency checker verifies that the text of labels conforms to OpNom.
  - **Used OpNom language**: Determines which language dictionary to use in addition to the OpNom dictionary. English words in the selected dictionary will not be flagged by the consistency checker.
  - **Used OpNom check**: Toggles the ESA Annex to the OpNom on/off.
- **Data Source**: Context setting for the data source editing, defines which contexts shall be possible and which are the default context.
  - **Context List**: Determines the list of contexts used by the editor, mainly in the data source dialog, but also during drag-and-drop operations from the System Configuration Browser.

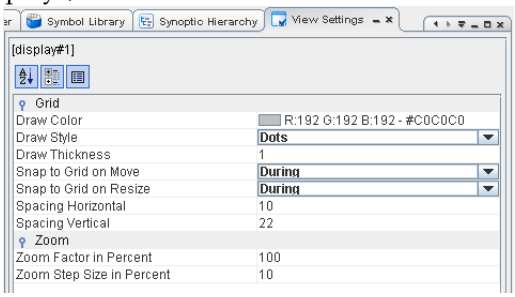
  - **Default Context**: Defines which context is the default one from the list above, used mainly in data source dialog.

- **Data Source**: Definition of the **Context List** which is the set of TM identifiers e.g. Opsname, Pathname, SID, PUI. The Context list is used in data source dialogs and the Label Context property. **Context default** is the context that is shown in Context selection combo-boxes as the default entry.
- **SCOE**: SCOE file path
- **Autoload**: IMDB properties used to launch editor (Optional)

The preferences are stored in the uss.config file which should not be modified outside the Editor. For information on how to configure the Editor in the MCS environment, see USS Editor Parameter.

4.11.3 View Settings

The View Settings property pane can be accessed from the View menu. The View Settings present and allow modification of the grid and zoom settings for the current display. In order to change grid and zoom default settings for new displays, use the Preferences.



View setting

Chapter 5

Executor

5.1 Introduction

USS provides a synoptic display execution environment for on-board and ground application. The executor can be run on ground based computers using LINUX, SOLARIS and MS Windows. The executor executes displays which have been authored with the editor.

This chapter explains how you can configure the executor as you wish e.g. by enabling tooltips, and how you can exit the executor. Starting the executor is explained in Getting Started.

5.1.1 Configuring User Settings

5.1.1.1 Prerequisites

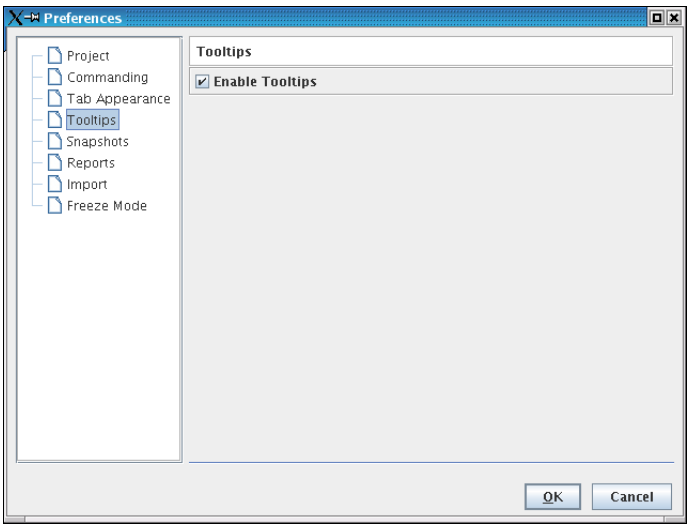
- Executor must be running.

5.1.1.2 Configuring User Settings

This section gives a short general quick start on how to configure user settings. All available preferences and their corresponding dialogs are explained in full detail in Configuring the Executor.

To view and change your personal executor application settings:

1. Choose **Options > Preferences...** .
2. In the popup window, select what you want to configure. (E.g. select **Tooltips** in the tree structure on the left side.) Other choices would be Project, Commanding, Tab Appearance, Snapshots, Reports, Import or Freeze mode.



3. Change the settings (e.g. Check or uncheck the checkbox to enable/disable tooltips.).
4. Click **OK**.

5.1.2 Exiting the Executor

5.1.2.1 Prerequisites

- Executor must be running.

5.1.2.2 Exiting the Executor

To exit the executor:

1. Choose **File > Exit**.
2. Executor exits. The current window layout is saved and can be restored on the next run.

5.1.2.3 See also

- Starting the Executor

5.2 Monitoring and Control Configuration

5.2.1 Connecting to System to be Monitored and Controlled

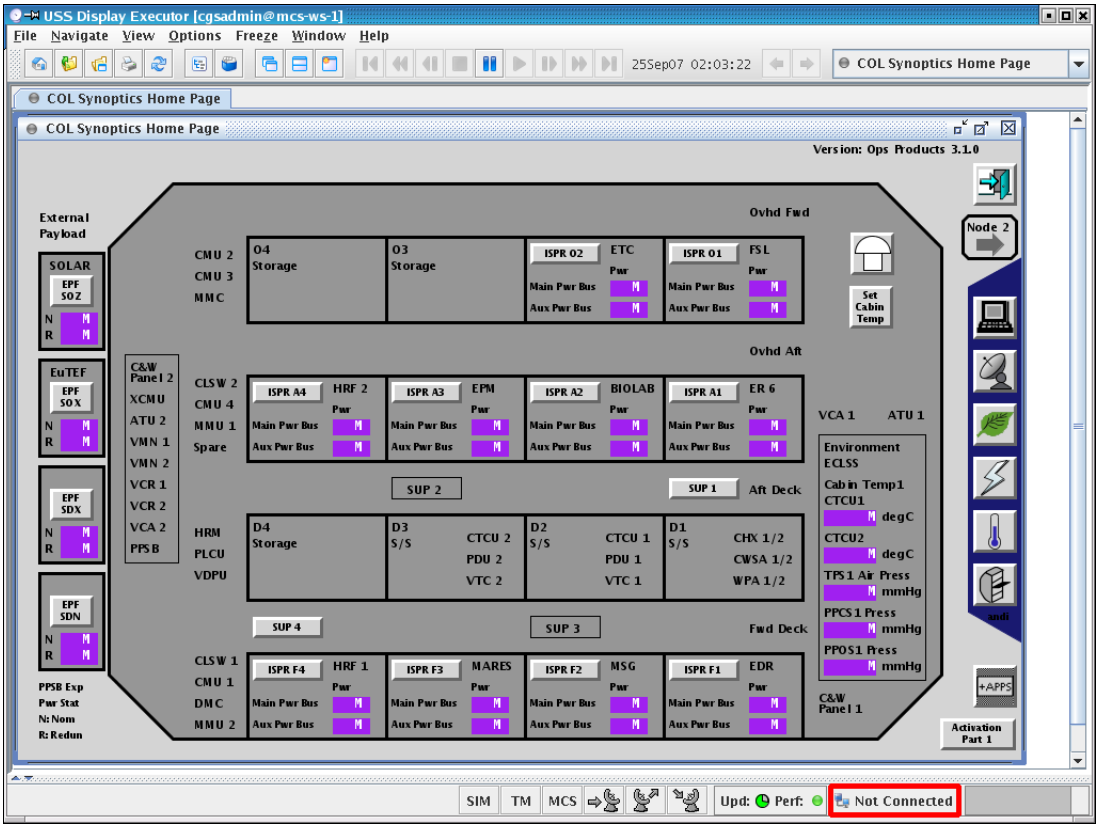
5.2.1.1 Prerequisites

- Executor must be running.

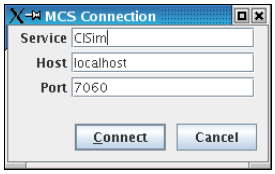
5.2.1.2 Connecting to System to be Monitored and Controlled

To establish data connection with the system to be monitored and controlled:

1. Do one of the following:
   - Choose **Options > Connection...** .
   - Click the **Not Connected** button in the status-bar.



2. In the popup dialog change the connection parameters as needed.



3. Click **Connect.**

Connection to MCS/CIS established (reflected by executor **status indicator**, i.e. former gray icons turn green and button in status-bar which switches from Not connected button into **Connected** button.

5.2.1.3 See also

- Disconnecting System to be Monitored and Controlled

5.2.2 Disconnecting System to be Monitored and Controlled

5.2.2.1 Prerequisites

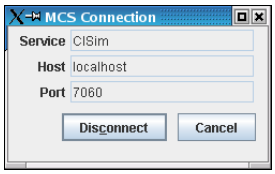
- Connection must have been established already.

5.2.2.2 Disconnecting System to be Monitored and Controlled

To stop data connection with the system to be monitored and controlled:

1. Do one of the following:
   - Choose **Options > Connection...** .
   - Click the **Connected** button in the status-bar.

2. The popup dialog appears.



3. Click **Disconnect**.
4. In the log window the text **"Disconnected from CIS"** is displayed.

Disconnected from CIS (reflected by executor status indicator, which turns from green back to gray and the text of the button now again shows **Not Connected**).

5.2.2.3 See also

- Connecting System to be Monitored and Controlled

5.2.3 Switching Target for Commands

5.2.3.1 Switching Target for Commands

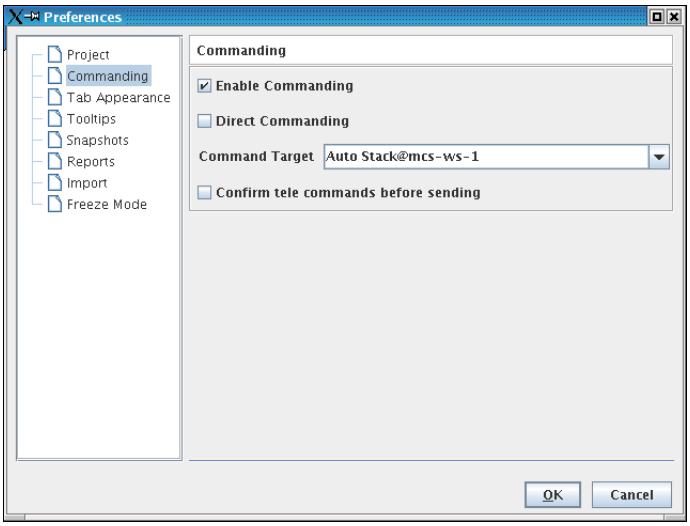
You can switch between different targets for the commands. The available targets are:

1. direct command string to CIS,
2. stack either manual or auto stack,
3. local log file/window as command target.

The using of local log file/window as command target turns effectively the sending of remote commands off.

To switch target for commands:

1. Choose **Options > Preferences...** .
2. In the popup dialog select **Commanding**.
3. Check **Enable Commanding**.
4. Uncheck **Direct Commanding**.
5. From drop-down list choose new **Command Target** (e.g. Manual or Auto Stack).



6. Check **Confirm tele commands before sending** if you wish commands to be confirmed.
7. Click **OK**.

From now on, all commands are directed to the new target.

5.2.4 Checking Acquisition State

5.2.4.1 Prerequisites

- Connected to monitored system.

5.2.4.2 Checking Overall Acquisition State

USS provides an indication of acquisition status. The executor gives indicators on the update performance indicator. In case of severe problems the user is notified. User notification is done via dialogs and/or a log window.

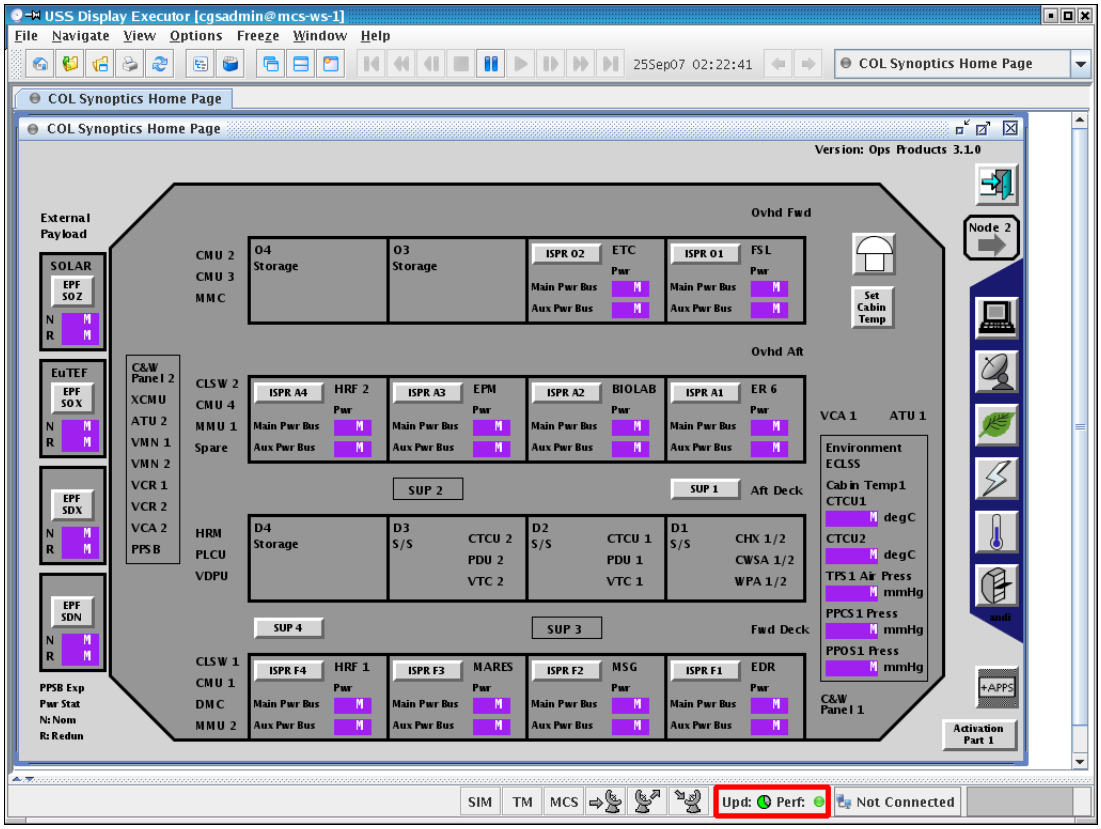
You can verify:

- that a display still gets updates from the monitored target system, and
- that the display executor software is functioning (i.e. the display is not frozen due to local software failure).

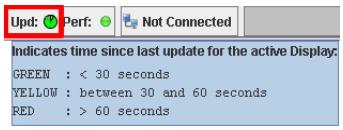
as described below.

To check overall acquisition state:

1. Select display to check.
2. In the status-bar two fields show continuously updated icons.



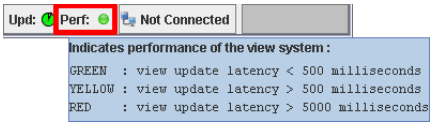
- **Update:** The first item is a small clock icon with a needle moving once a second. Every time a parameter in the display is updated the timer is reset and starts over again. The clock indicates the time since the last update for the active display. The color of the icon changes depending on when the last update of the selected display has been. Green means it has been updated within the last 30 seconds.



If the last update was between 30 and 60 seconds the icon is yellow.

If the display hasn't been updated for more than one minute the icon turns red.

- The second and last item is the **performance indicator**. It indicates the performance of the view system. If the update latency is less than 500 milliseconds the icon is green. If the latency is bigger than 500 milliseconds the color changes to yellow. Red means that the update latency is bigger than 5000 milliseconds, which means that some data may be dropped.

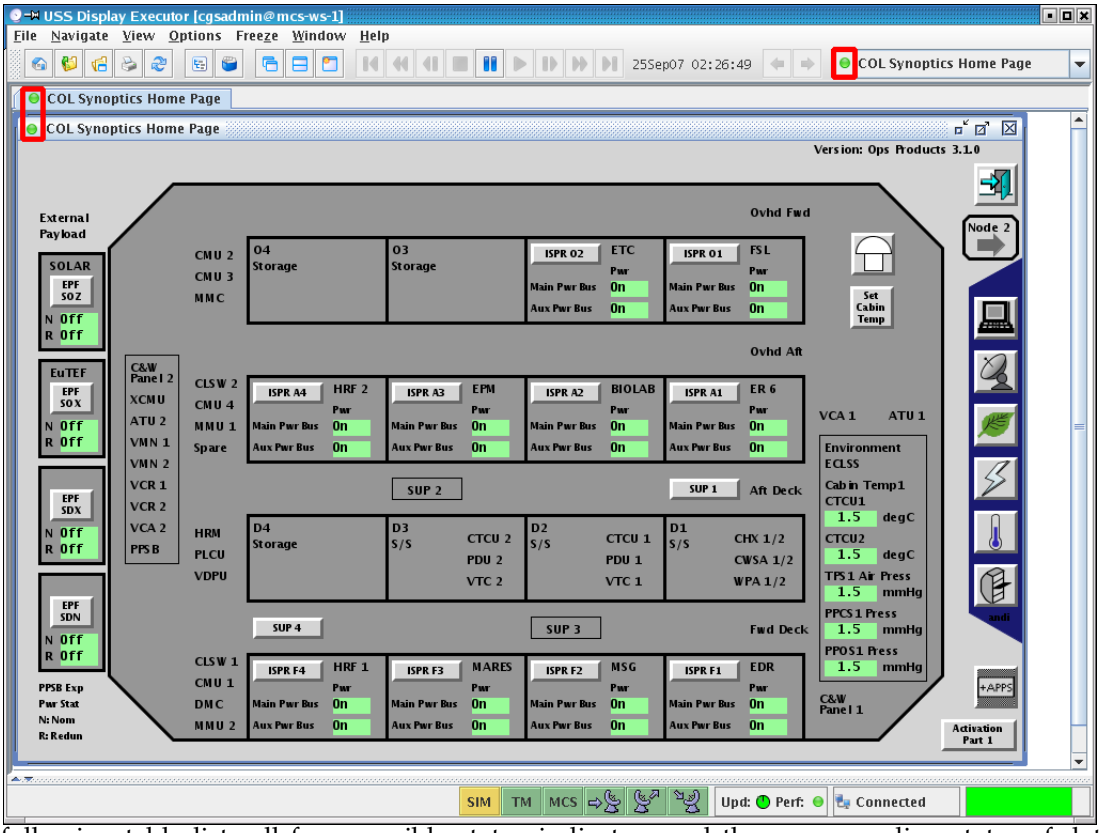


5.2.5 Checking Monitoring State

This section describes how the **monitoring state** can be viewed for a **single** display or for **all** open displays.

5.2.5.1 Display Status Indicator

Each display has a coloured indicator that changes colour according to the state of its data sources.



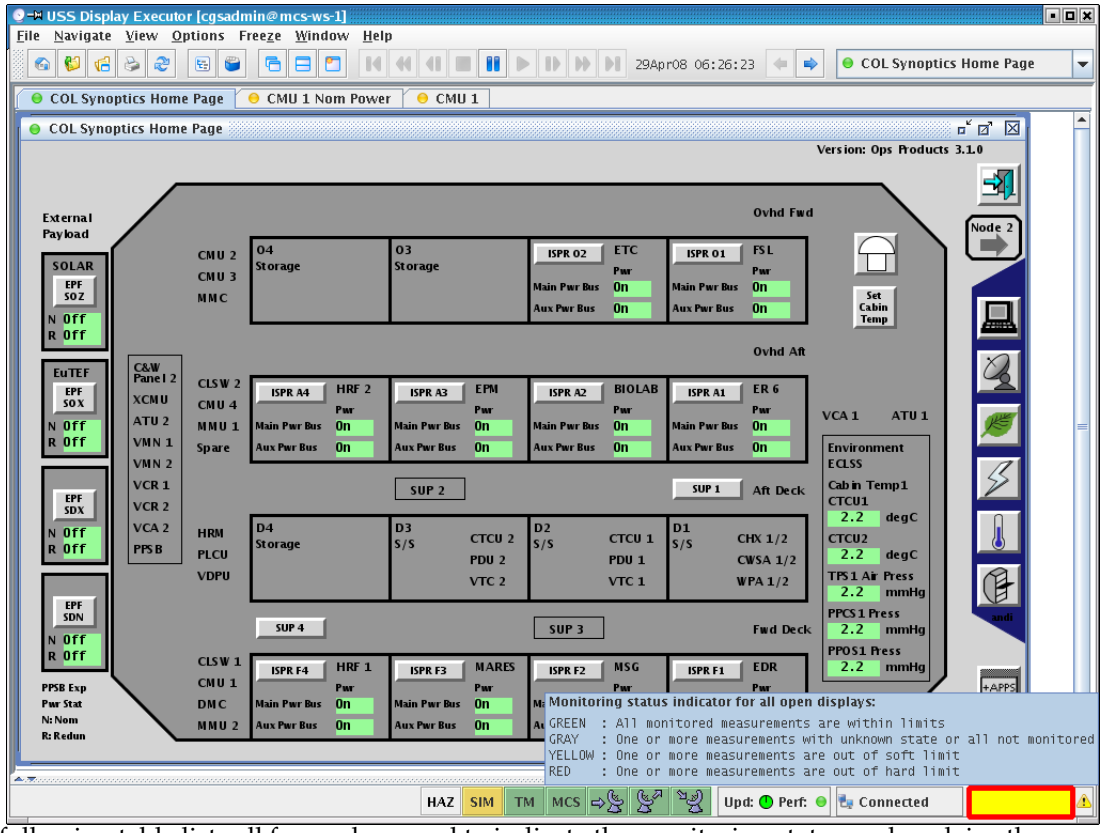
The following table lists all four possible status indicators and the corresponding states of data sources in the display.

Table 5.1: Indication of Acquisition Status for Status of Data Sources

Display Status Indicator	Status of Data Sources
Display Status indicator is not displayed at all.	Display contains no data sources.
Display Status indicator is grey.	For each data source in the display the status is "not monitored".
Display Status indicator is green.	At least one data source is "in limits" and no data source has a caution (soft/nominal) or warning (danger) limit violation.
Display Status indicator is yellow.	At least one data source has a caution (soft/nominal) limit violation and no data source has a warning (danger) limit violation.
Display Status indicator is red.	At least one data source has a warning (danger) limit violation.

5.2.5.2 Overall Monitoring Status Indicator

The overall monitoring status indicator shows the monitoring status for all currently open displays.


USS Display Executor [cqsadmin@mcs-ws-1]
File Navigate View Options Freeze Window Help
COL Synoptics Home Page
CMU 1 Nom Power
CMU 1
Version: Ops Products 3.1.0
Ovhd Fwd
Ovhd Aft
Fwd Deck
Aft Deck
Monitoring status indicator for all open displays:
GREEN : All monitored measurements are within limits
GRAY : One or more measurements with unknown state or all not monitored
YELLOW : One or more measurements are out of soft limit
RED : One or more measurements are out of hard limit
HAZ
SIM
TM
MCS
Upd:
Perf:
Connected

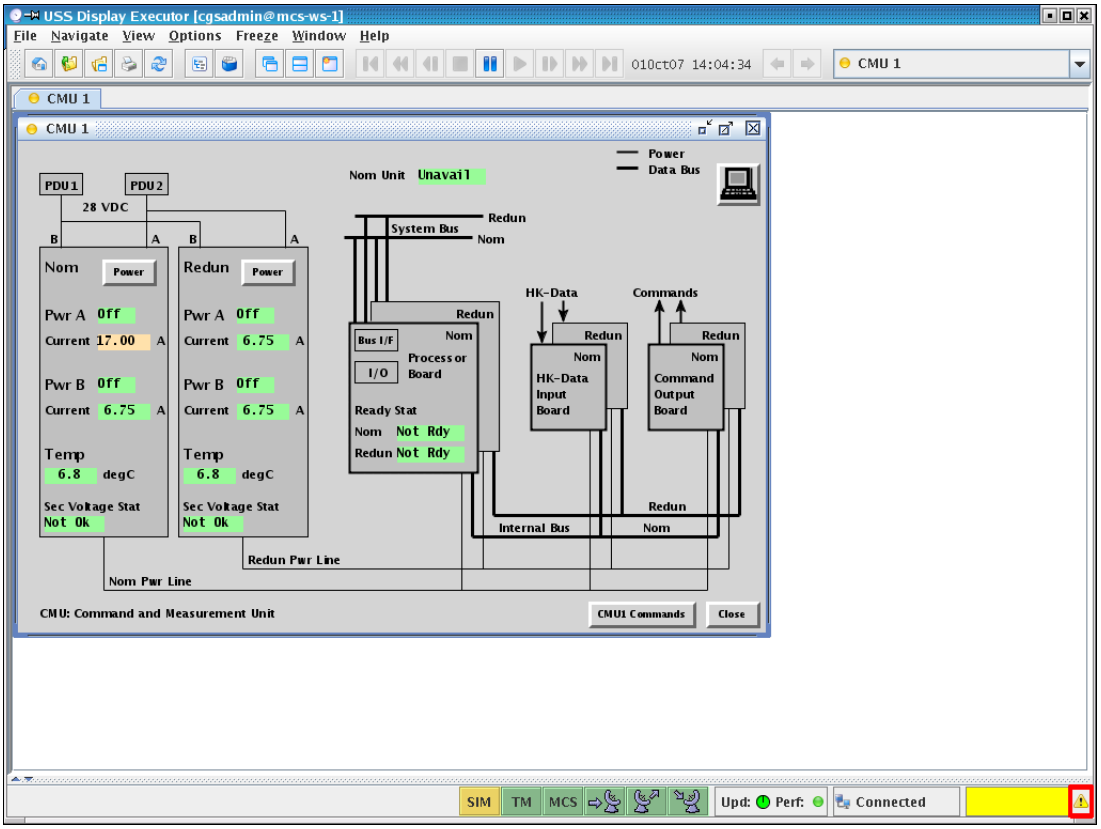
The following table lists all four colors used to indicate the monitoring status and explains the corresponding overall monitoring state. The colors given in the table are sorted by criticality, beginning with the weakest and ending with the highest criticality.

Table 5.2: Overall monitoring status for all open displays

Overall Monitoring Status Indicator	Status of Measurements
Green	All monitored measurements are within limits.
Grey	One or more measurements with unknown state or all not monitored.
Yellow	One or more measurements are out of soft limit.
Red	One or more measurements are out of hard limit.

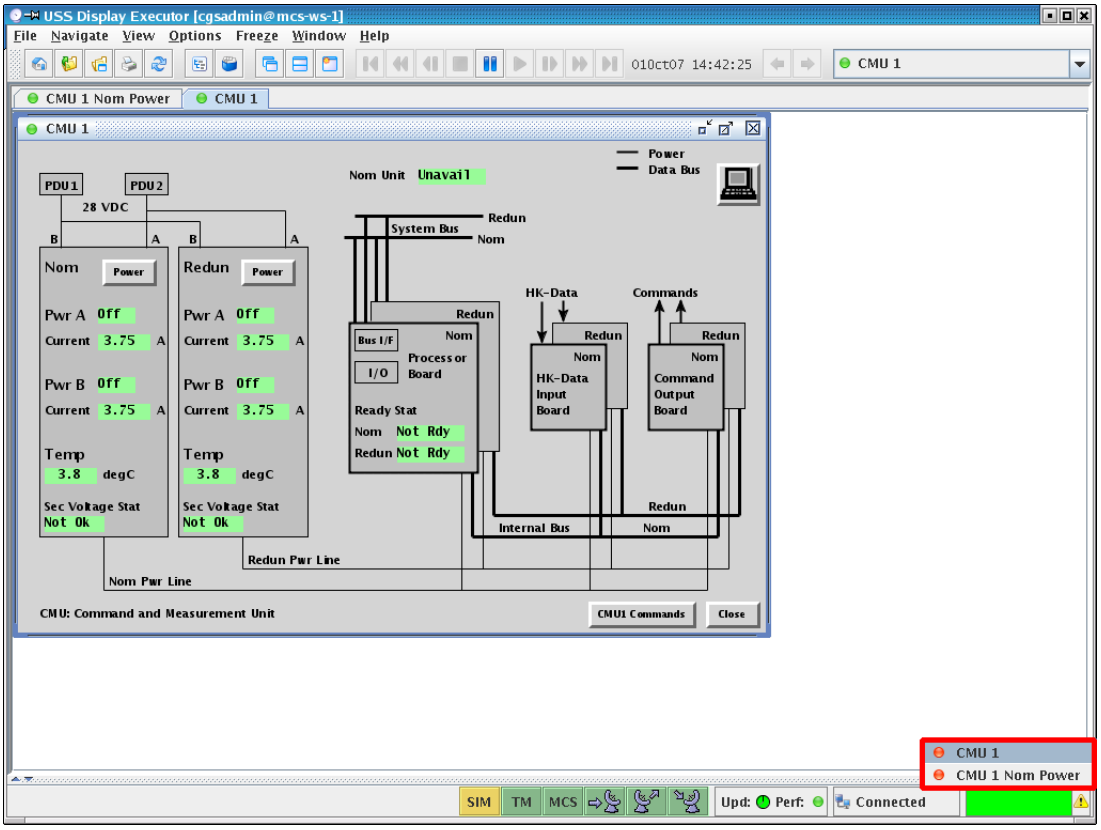
The color used is always that of the highest criticality (i.e. red, yellow, grey and green), even though the current status may be less severe (as demonstrated in the screenshot above, where the overall monitoring status indicator is yellow, even though the status of the displays is green and yellow).

5.2.5.2.1 Monitoring status history All displays that had or have out-of-limit measurements are maintained in a history. If the history has entries a yellow triangle is displayed on the right hand side of the overall monitoring status indicator. When the history is empty the yellow triangle is not displayed.



To show monitoring status history:

1. Right-click on the yellow triangle on the right hand side of the overall monitoring status indicator.
2. In the popup menu select the display you want to navigate to.
3. The selected display is removed from history.

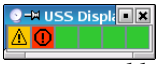


Displays whose measurements are within limits again, are still kept in history. Displays are kept in the history until display has been selected as explained above. That means no display is automatically removed from the history. The only way to remove a display from the history is through user interaction.

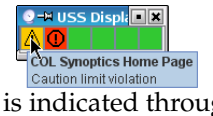
5.2.5.3 Display Status Overview Window

The display status overview presents the status of all open displays in a separate window. In this window the status of each display in the executor is represented by a button.

To open the display status overview Window select **View > Status Overview**



A click on a status button brings the display represented by the button to the foreground. To just get the title for a display represented by a button move the mouse over the button and wait until the tooltip is displayed as shown below:



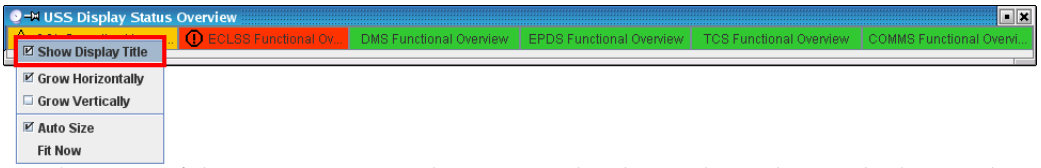
The overall monitoring status of a display is indicated through the buttons color and icon as shown in the following table.

Table 5.3: Display status overview monitoring status

Display status overview button	Status of Measurements
	All monitored measurements are within limits.
	One or more measurements with unknown state or all not monitored.
	One or more measurements are out of soft limit.
	One or more measurements are out of hard limit.

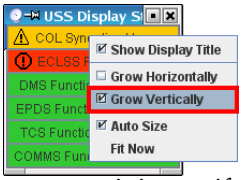
The buttons can be configured to also show the display title. To show the display title in all display status overview buttons:

- Right-click on a button in the display status overview window.
- Select **Show Display Title** from the pop-up menu.



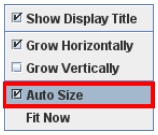
The layout direction of the status overview buttons can be changed. To change the layout direction to vertical do the following:

- Right-click on a button in the display status overview window.
- Select **Grow Vertically** from the pop-up menu.



The size of the window is automatically kept to a minimum if the **Auto Size** option is enabled. If this option is turned off the window can be re-sized manually. To toggle the auto size option for the window do the following:

- Right-click on a button in the display status overview window.
- Select **Auto Size** from the pop-up menu.



After a change of the position or size of the display status overview window it can be necessary to trigger a layout of its contents. To let the window adapt to size or position changes do the following:

- Right-click on a button in the display status overview window.
- Select **Fit Now** from the pop-up menu.

5.2.5.3.1 Configuring the Display Status Overview Window

5.2.5.3.1.1 Enable the Display Status Overview Window on Executor startup To show the display status overview window on Executor startup do the following:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the following property to true.

```
### Set to true to make the display status overview window visible at application
### startup.

uss.executor.display.status.overview.enabled=false
```

5.2.5.3.1.2 Set the inital Position of the Display Status Overview Window To set the initial screen position of the display status overview window do the following:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the x and y position to the desrired values.

```
### Define the initial x and y position of the display status overview frame.
### The default position is 0, 0.

uss.executor.display.status.overview.window.x=0
uss.executor.display.status.overview.window.y=0
```

5.2.5.3.1.3 Set the inital Layout Direction of the Display Status Overview Window To set the initial layout direction of the display status overview window do the following:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the following property to true for horizontal layout or to false for vertical layout.

```
### Set the value to true if the status indicators in the display status overview ↩
     frame
### should be layouted horizontally, otherwise the indicators are layouted  ↩
    vertically.
### The default value is true.

uss.executor.display.status.overview.layout.horizontal=true
```

5.2.5.3.1.4 Enable the Display Title on the Status Buttons To enable the display title on the status buttons do the following:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the following property to true.

```
### Set the value to true if the title of the display should be displayed in
### the status indicators of the display status overview frame.
### The default value is false.

uss.executor.display.status.overview.showtitle=false
```

5.2.5.3.1.5 Configuring the Button Colors You can configure the button colors and icons for the overall display monitoring states.

To configure the color for the monitoring states:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the desired color for the corresponding monitoring status.

```
### Possible values to configure are:
### NO_PARAMETERS        Display contains no parameters
### FROZEN               Display is in freeze mode
### OK                   All parameters are monitored and in limits
### UNKNOWN              No Status information available
### ALL_NOT_MONITORED    All parameters have a non-nominal acquisition status
### OUT_OF_LIMIT_CAUTION At least one parameter exceeds caution limits
### OUT_OF_LIMIT_WARNING At least one parameter exceeds warning limits

uss.display.status.color.noparameters=#C0C0C0
uss.display.status.color.frozen=#C0C0C0
uss.display.status.color.ok=#32C832
uss.display.status.color.unknown=#C0C0C0
uss.display.status.color.allnotmonitored=#C0C0C0
uss.display.status.color.outoflimitcaution=#FBC800
uss.display.status.color.outoflimitwarning=#FF3200
```

5.2.5.3.1.6 Configuring the Button Icons You can configure the icons to be displayed in the display status buttons for the out of limit monitoring states.

To configure the icons for the out of limit monitoring states:

- Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the desired icon for the corresponding monitoring status.

```
### Possible values to configure are:
### OUT_OF_LIMIT_CAUTION At least one parameter exceeds caution limits
### OUT_OF_LIMIT_WARNING At least one parameter exceeds warning limits

uss.display.status.icon.outoflimitcaution=${basedir}/share/icons/status/caution. ↩
    png
uss.display.status.icon.outoflimitwarning=${basedir}/share/icons/status/warning. ↩
    png
```

5.3 Window Handling

5.3.1 Saving Window Layout

5.3.1.1 Prerequisites

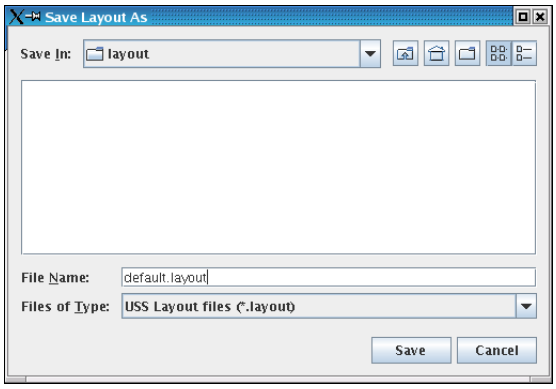
- At least one display must be open, otherwise the menu item is disabled.

5.3.1.2 Saving Window Layout

You can save the layout of all displays including size and position so that the window layout can be restored later.

To save window layout:

1. Choose **File > Save Layout As...** .
2. In the File chooser dialog, select the folder in which you want to save the layout (e.g. folder layout).
3. In the file name text area enter the name under which you want to save the layout.



4. Click **Save**.

5.3.1.3 See also

- Loading Window Layout

5.3.2 Loading Window Layout

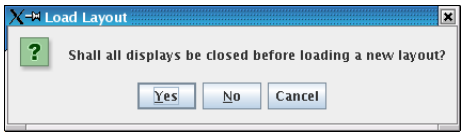
5.3.2.1 Prerequisites

- Window layout must exist.

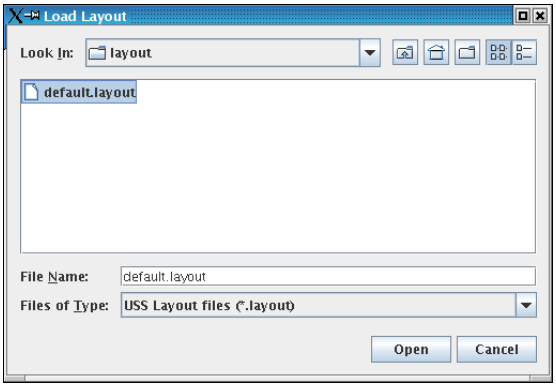
5.3.2.2 Loading Window Layout

To load a previously saved window layout:

1. Choose **File > Load Layout...** .
2. If there are already any displays open you'll be asked if you wish to close all open displays before loading a new layout. Select either **Yes, No or Cancel**. If you chose **No** a display which is already open and is also contained in the layout will be opened twice. So if you don't want any display to be opened more than once click **Yes**.



3. In the File chooser dialog, select the layout you which to load (e.g. default.layout).



4. Click **Open**.

The current layout is replaced with the selected layout.

5.3.2.3 See also

- Saving Window Layout

5.3.3 Loading Display from File System

5.3.3.1 Prerequisites

- Window layout must exist.

5.3.3.2 Loading Display from File System

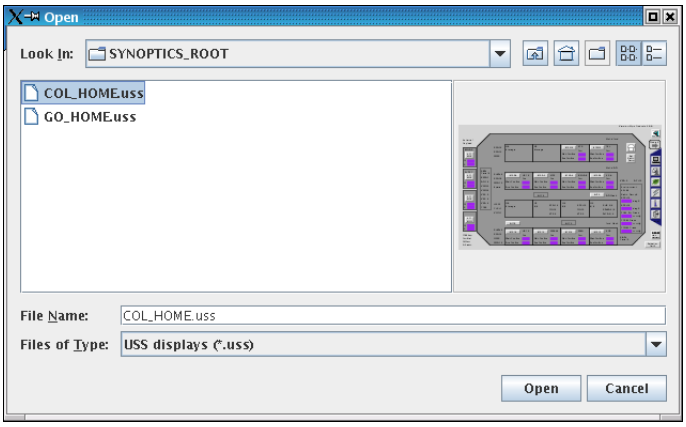
USS supports the displaying of at least 20 synoptic displays in parallel. A display can be loaded multiple times.

The file hierarchy is the directory tree of the file system e.g. `/examples/uss/fwdu`, where the directories are the nodes and the displays are the leafs of the hierarchy structure. The displaying of the file name depends if it is a PCS, FWDU or GWDU display. The file name of a PCS display is defined by the NASA. The file name of a FWDU display is its MDB end item name, which is defined by the display author when storing the display into the MDB. The same naming rules apply to GWDU displays. In the headline of the window the displays' opsname is displayed not its file name. If you want to view the file name you must choose File > Properties as described in chapter Show display properties.

To load display from file system:

You can load a saved display from the file system as described below.

1. Choose **File > Open...** .
2. In the File chooser dialog, select the display you which to load. A preview will be displayed on the right hand side.



3. Click **Open**.

The display is loaded and opened. No check for duplicate displays is made, i.e. a display can be loaded multiple times.

5.3.4 Reloading Display from File System

5.3.4.1 Prerequisites

- Make sure that the display is currently selected.

5.3.4.2 Reloading Display from File System

You can reload a display from file system after changes have been made to the file e.g. with the editor.
To reload display from file system:

1. Select Display by clicking on its tab. The selected tab changes to a different color.
2. Choose **File > Reload**.

Display is reloaded from file system.

5.3.5 Resizing Display Window

5.3.5.1 Prerequisites

- There must be at least one display open.

5.3.5.2 Resizing Display Window

USS allows to resize a synoptic display. The contents of the display such as fonts and graphics are rescaled accordingly.
To resize display window:

- Click and drag window border.

5.3.6 Resetting Display Window to Default Size

5.3.6.1 Prerequisites

- There must be at least one display open.

5.3.6.2 Resetting Display Window to Default Size

You can reset a display window to its predefined default size.
To reset display window to default size:

- Choose **Window > Reset to default size** .

The display window is reset to its default size.

5.3.7 Showing/Hiding Command Responses

5.3.7.1 Prerequisites

- There must be at least one display open which contains tele commands.

5.3.7.2 Showing/Hiding Command Responses

USS allows to hide the command responses for a display containing tele commands.
To hide/show command responses:

- Choose **Window > Show Command Response**.

5.3.8 Navigating Display Hierarchy

5.3.8.1 Navigating Display Hierarchy

USS provides the capability to navigate between different displays. You can navigate through the display hierarchy in order to find, open or select a display. The display hierarchy mirrors the subsystem structure in which displays are organized. You can open another synoptic from the synoptic tree visualization or from navigation buttons. Navigation buttons can also have the property to specify that the current window will be replaced on open.

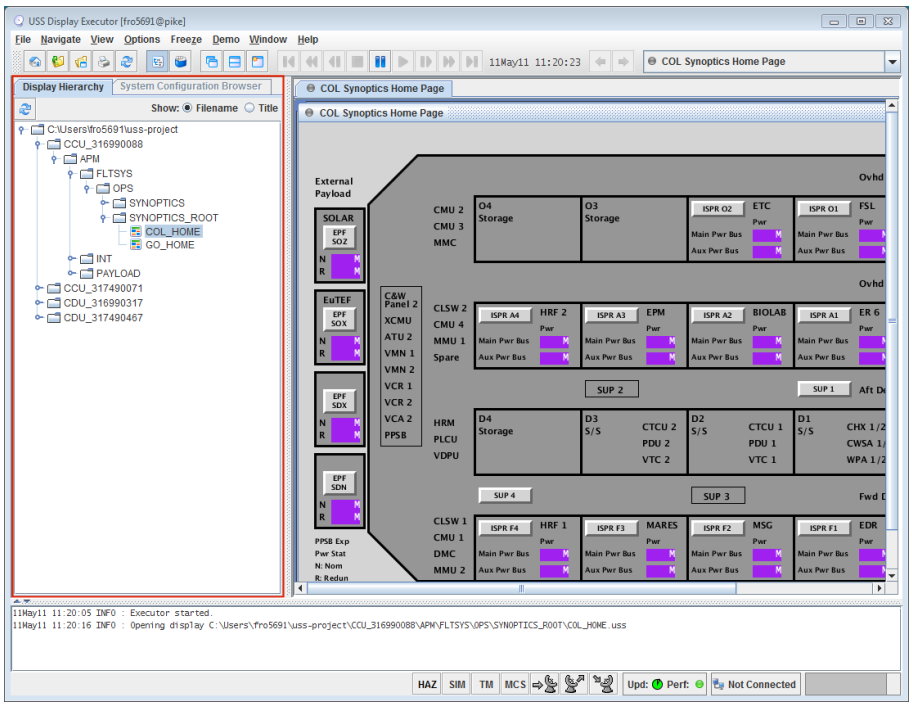
Navigation in this context means nothing else then exploring any set of displays. Where displays are selected via point and click. They are automatically loaded.

To navigate display hierarchy:
Do one of the following:

- Navigating via hierarchy panel:

  1. Choose **View > Show Browser**, if browser isn't already visible.
  2. The display hierarchy (see red rectangle in next screenshot) is shown as a tree panel on the left side of the workspace.
  3. Select the display hierarchy tab, if hierarchy isn't already visible.

     Click into hierarchy panel. Select Display. The selected display opens. If it was already open, the selected display is focused, otherwise the display is loaded and opened.



- Navigating via buttons:

  – Navigate via buttons embedded in display by display author. The navigation strategy is defined by the display author.

5.3.9 Navigating System Configuration Browser

5.3.9.1 System Configuration Browser

USS provides the capability to navigate through the system configuration.

To navigate system configuration:
Do one of the following:

- Navigating via system configuration:

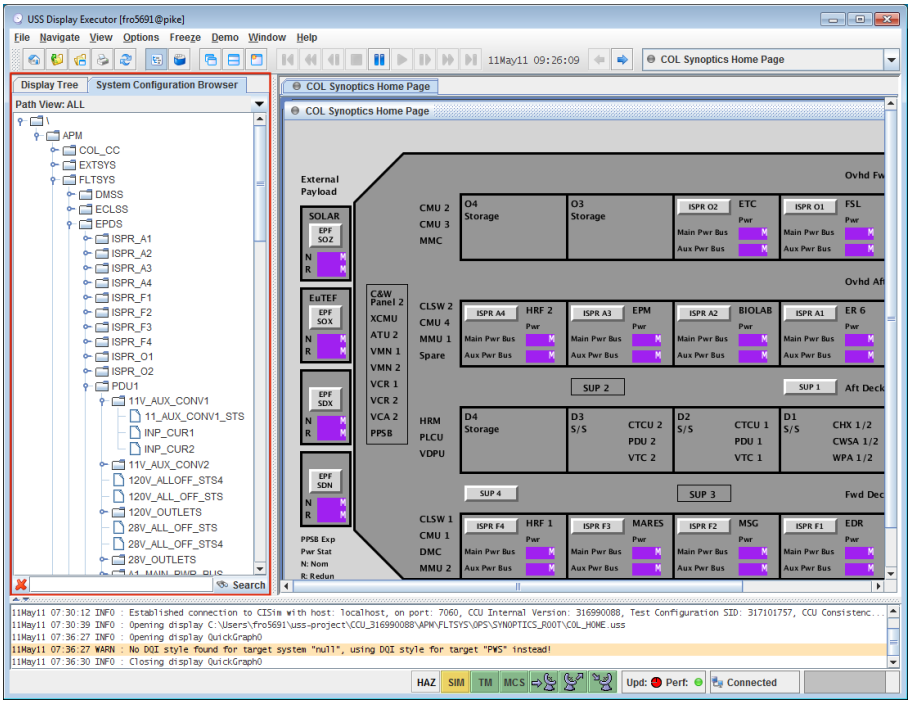
  1. Choose **View > Show Browser**, if the browser isn't already visible.

2. Select the system configuration tab, if system configuration browser isn't already visible. The system configuration tab is disabled if no configuration data are available for the selected CIS connection.

   The system configuration (see red rectangle in next screenshot) is shown as a tree panel on the left side of the workspace.

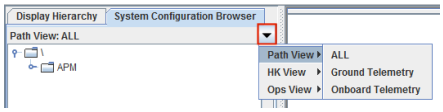
3. Click into system configuration panel and navigate through the system configuration. There are three views (PATH, HK, OPS) available to navigate through the system configuration and show an item as quick graph.

   The next chapters describes how to select the views and show an item as quick graph.



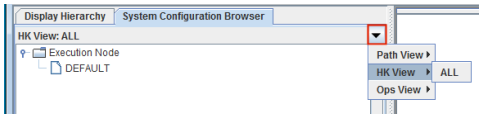
5.3.9.2 System Configuration: Path View

Select the path view and one of its filter (see next screenshot) to have a view to the system configuration path data.



5.3.9.3 System Configuration: Hk View

Select the HK view and one of its filter (see next screenshot) to have a view to the housekeeping data.



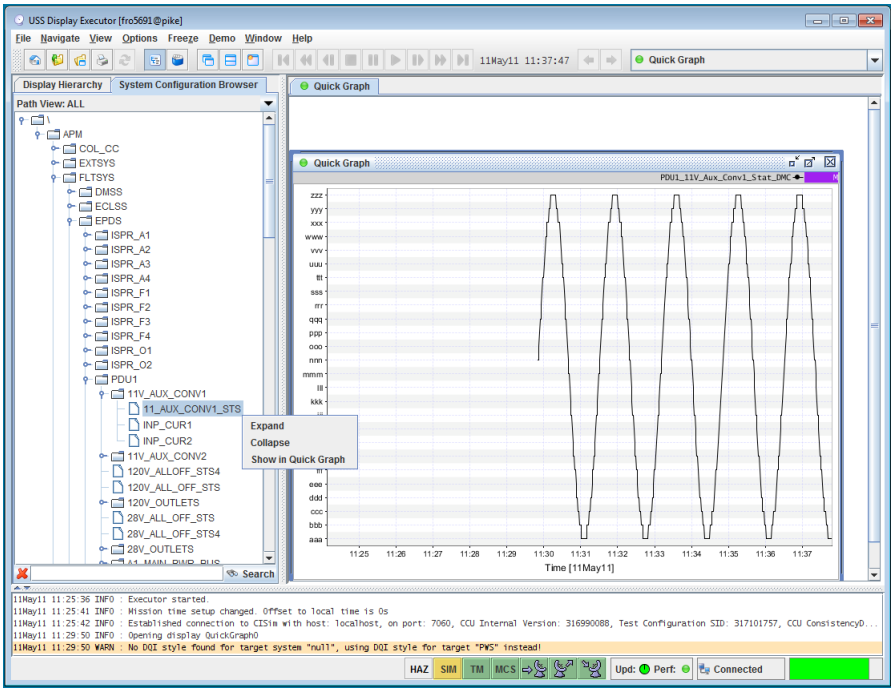
5.3.9.4 System Configuration: Ops View

Select the OPS view and one of its filter (see next screenshot) to have a view to the operating system data.

5.3.9.5 System Configuration Browser: Quick Graph Menu

Select an item in the system configuration tree and press the right mouse button. If the selected item can be shown in the quick graph dialog (see next screenshot) the menu entry 'Show in Quick Graph' is enabled.

Select the menu item to start the quick graph dialog.



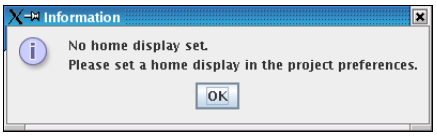
5.3.10 Navigating to Home Display

5.3.10.1 Prerequisites

- Home display must be configured via **Options > Preferences...** . Select **Project** tab on left hand side. Either click **Use selected display** to set current active display as home display or click **Browse...** button in border area labeled **Home Display**.

5.3.10.2 Navigating to Home Display

USS provides the capability to navigate between different displays. The standard or default home display is the **COL Synoptics Home Page** under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS_ROOT/COL_HOME.uss`. If the home display isn't configured as explained above under prerequisites following error message will appear.



To navigate to home display:
Do one of the following:

- Click on the **homepage button** in the toolbar.

USS Display Executor [cgsadmin@mcs-ws-1]
File Navigate View Options Window Help
00:00:00.000
COL Synoptics Home Page (COL_Home_P

- Choose **Navigate > Home**.

Home display is opened.

5.3.11 Showing/Hiding the Toolbar

5.3.11.1 Prerequisites

- Executor is running and toolbar is visible.

5.3.11.2 Showing/Hiding the Toolbar

USS provides the capability to hide the toolbar to provide more space for the display view.

- Choose **View > Toolbar**, if toolbar isn't already hidden.

Toolbar is hidden.

5.3.12 Closing Display

5.3.12.1 Prerequisites

- Display must be open and selected.

5.3.12.2 Closing Display

Closes only one display: the currently selected display.
To close display:

- Select Display you wish to close. The tab is displayed in a different color.
- Do one of the following:
  - Choose **File > Close**.
  - Right-click on tab select **Close**.

Display is closed.

5.3.12.3 See also

- To close all open displays: Closing All Displays
- To close all displays but the currently selected display: Closing other displays

5.3.13 Closing All Displays

5.3.13.1 Prerequisites

- At least one display must be open.

5.3.13.2 Closing All Displays

Closes every open display.
To close all displays:

- Choose **File > Close all**.

All displays are closed.

5.3.13.3 See also

- To close only one display which must be selected: Closing Display
- To close all but the currently selected display: Closing Other Displays

5.3.14 Closing Other Displays

5.3.14.1 Prerequisites

- At least one display must be open.

5.3.14.2 Closing Other Displays

Closes all open displays but the selected one. So at the end there's only one display left: the currently selected display. It isn't possible to select more displays to be left open.

To close other displays:

- Do one of the following:
  - Choose **File > Close Other**.
  - Right-click on tab select **Close Other**.

All displays but the selected one are closed.

5.3.14.3 See also

- To close only one selected display: Closing Displays
- To close every open display: Closing All Displays

5.3.15 Toggling Tabbed Mode

5.3.15.1 Toggling Tabbed Mode

Displays can be shown either on a tabbed window (one window at a time) or as iconized internal windows (many in parallel).

To toggle tabbed mode:

- Choose **Window > Tabbed mode**.

GUI shows displays according to new tabbed mode.

5.3.16 Undocking Windows

5.3.16.1 Prerequisites

- At least one display must be open and selected.

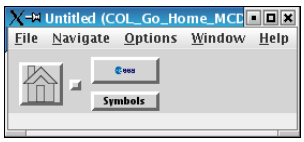
5.3.16.2 Undocking Windows

USS executor allows to undock windows from the executor so that they open in an external standalone window. All or just a single undocked window can be docked back into the executor. Only one window at a time can be undocked, so there's no multiple undocking although multiple docking is possible.

To undock windows:

- Choose **Window > Undock** in the executor.

The selected display is opened in an external standalone window.



5.3.17 Docking Windows

5.3.17.1 Prerequisites

- At least one window must have been undocked.

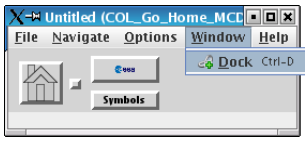
5.3.17.2 Docking Windows

Undocked windows which are displays which have been opened in an external standalone window can be docked back into the executor. Either a single display can be docked back or all undocked windows can be docked back into the executor. But there's no multiple undocking, though.

5.3.17.2.1 Docking a Single Window You can dock a single selected undocked window back into the executor.

To dock a single window:

- Select the undocked window.
- Choose **Window > Dock** in the undocked (that is external standalone) display.



The undocked window is docked back into the executor.

5.3.17.2.2 Docking All Windows You can dock all undocked windows back into the executor.

To dock all windows:

- Choose **Window > Dock All** in the executor.

All undocked windows are docked back into the executor.

5.4 Display Interaction

5.4.1 Showing Tooltip for Element

5.4.1.1 Showing Tooltip for Element

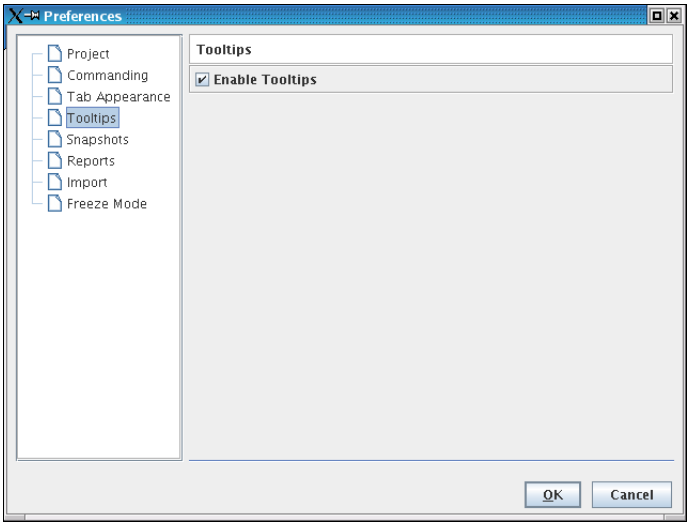
You can see tooltips associated with the element. You can turn tooltips globally on or off. In synoptic displays USS shows the opsname for end items as defined in the MDB. Tooltips are available over data fields and in parameter properties dialogs.

To show tooltip for element:

- Move mouse cursor over element.
- Tooltip appears at mouse position.

To enable/disable tooltips:

- Choose **Options > Preferences...** .
- On the tree structure on the left side select **Tooltips**.
- Uncheck checkbox before enable tooltips to disable tooltips. Check checkbox to enable tooltips again.



- Click **OK**.

It is not possible to configure the amount of time over which the tooltip is displayed.

5.4.2 Showing Element Properties

5.4.2.1 Show Element Properties

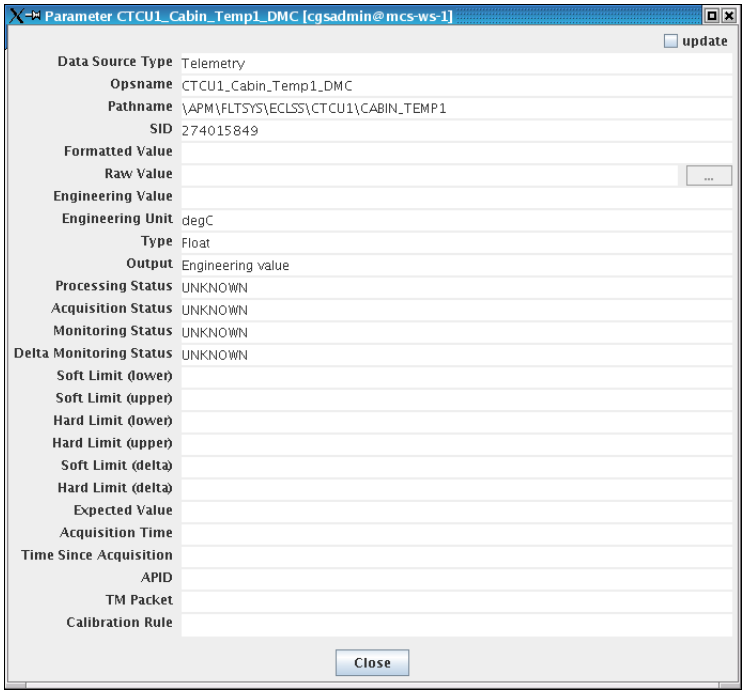
With show element properties you can get detailed information about TM/TC elements. USS provides a detailed information window on selection of display elements. The window shows the static and dynamic properties of the according parameters, e.g. parameter name (pathname, opsname), current parameter raw value, current parameter calibrated value. USS also provides a telemetry parameter query/debug popup.

To show element properties:

1. Right-click on any element (e.g. telemetry parameter f.e. **CTCU1_CABIN_TEMP1_DMC** in display **COL Synoptics Home Page** under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS_ROOT/COL_HOME.uss`).

2. Select **Properties** from element pop-up context menu.

3. A non-modal dialog appears with detailed information about the element properties. The dialog contents are updated as long as the dialog is open.



4. Click **Close**.

5.4.2.2 Showing Expression of Display

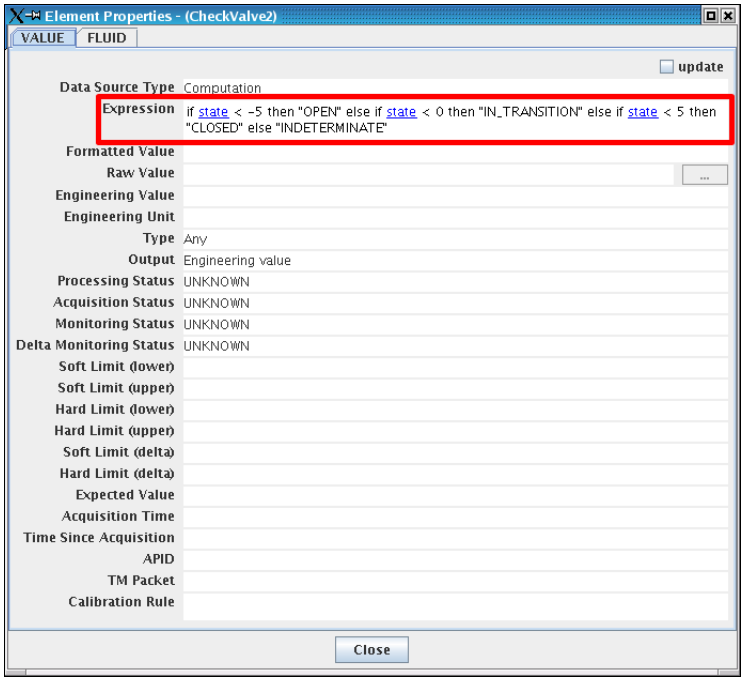
You can view the expressions of a display. Just right-click on any dynamic element e.g. rectangle, select Properties in the popup menu and in the popup window there'll be a table row called Expression in which the expression is listed.

To show expression of dynamic element:

1. Right-click on any dynamic element.

2. In the popup dialog select **Properties**.

3. The Parameter Information window will popup displaying the expression and other values.



4. Click **Close**.

5.4.3 Showing Display Properties

5.4.3.1 Prerequisites

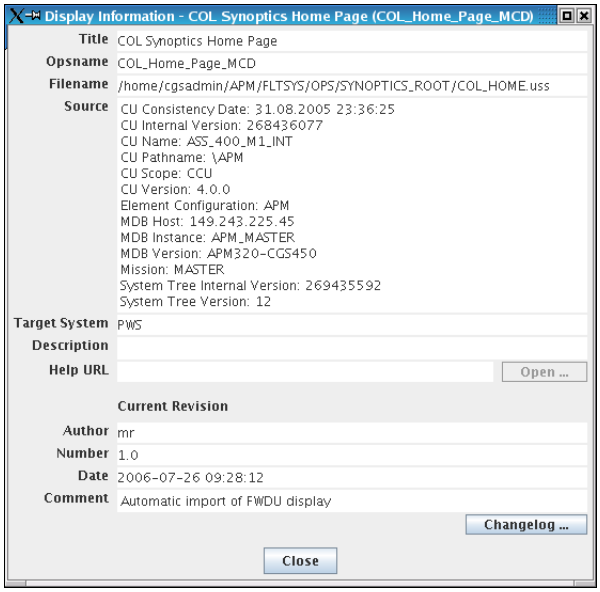
- Display must be open and selected.

5.4.3.2 Show Display Properties

USS provides several display properties including title, author, version and revison. Display properties also include a display comment, and change comments for each revision.

To get detailed information about a display:

1. Select **File > Properties**.
2. A non-modal dialog appears with detailed information about the selected display.



3. Click **Close**.

5.4.4 Copying Command to Clipboard

5.4.4.1 Prerequisites

- Display must be open and selected.

5.4.4.2 Copying Command to Clipboard

A command in this context is a command string as text. The content of the text depends on the underlying target system. You can copy the command to clipboard for use in other applications.

To copy command to clipboard:

1. Right-click over a command button. (E.g. **Pwr On** under label **PDU1 CTCU1 Pwr Bus** in display **COL Activation Part 1** under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ACTIVATION/ACT_PT_1.uss`).

2. Select **Copy Command to Clipboard** from popup context menu.

The opsname based command is copied to clipboard.

```
execute_flap (FLAP: PDU1_CTCU1_Pwr_Bus_On_AP(0),
ONBOARD_RECEPTION_NODE: \APM\FLTSYS\SW_SYS\CCSDS_CONFIG\END_NODES\DMC_DMS_SERV,
ONBOARD_EXECUTION_NODE: USM_SW_DMC_USS_Swop_Instance);
```

5.4.5 Copying Parameter Name to Clipboard

5.4.5.1 Prerequisites

- Display must be open and selected.

5.4.5.2 Copying Parameter Name to Clipboard

You can copy the parameter name to clipboard for use in another application.

To copy parameter name to clipboard:

1. Right-click over a parameter name. (E.g. parameter **CTCU1 Cabin Temp Setpoint** in display **Set Cabin Temp** under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ECLSS/CABIN_TEMP.fwdu`).

2. Select **Copy OPS Name to Clipboard** from popup context menu.

The opsname of parameter is copied to clipboard. For the above mentioned example the following text: **CTCU1_Cabin_Temp_Setpoint_DMC**.

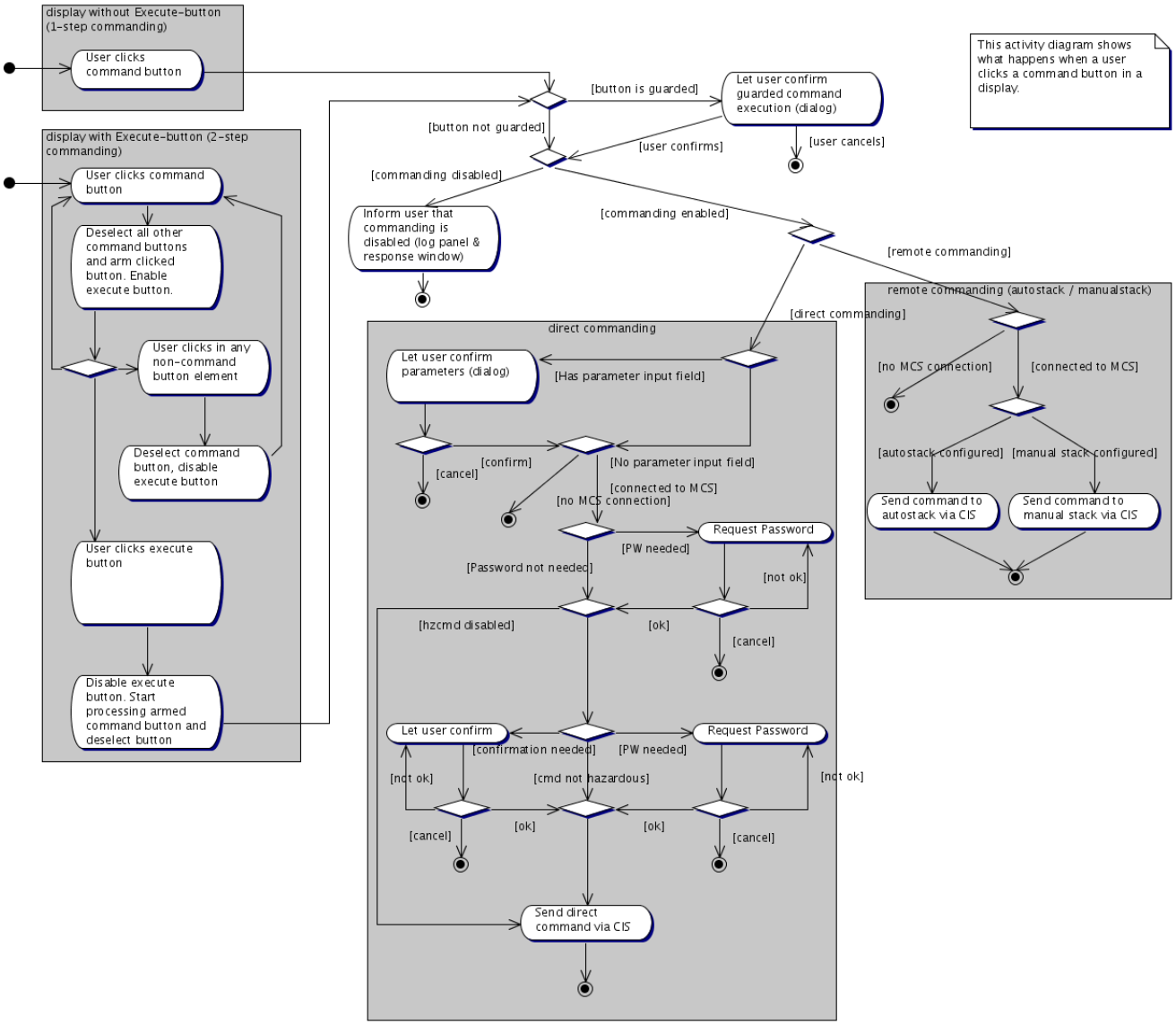
5.4.6 Issuing Telecommand via Command Button

5.4.6.1 Prerequisites

- Selected display contains command button.

5.4.6.2 Issuing Telecommand via Command Button

Commands can be sent in different ways. They can be sent in 1-step-commanding, this is done in displays without execute button. In displays with execute button they are sent as 2-step-commanding. Two step commands always require selecting two command buttons in order to execute the command to be sent, this is done by the execute button. Commands can then be send as direct or remote commanding. Direct commanding needs to be confirmed by the user, whereas remote commands are send without confirmation to auto-stack or manual-stack. The following activity diagram shows what happens when a user clicks a command button in a display.



USS allows for operator initiation of commands from synoptic displays. Command buttons are used to execute predefined commands.

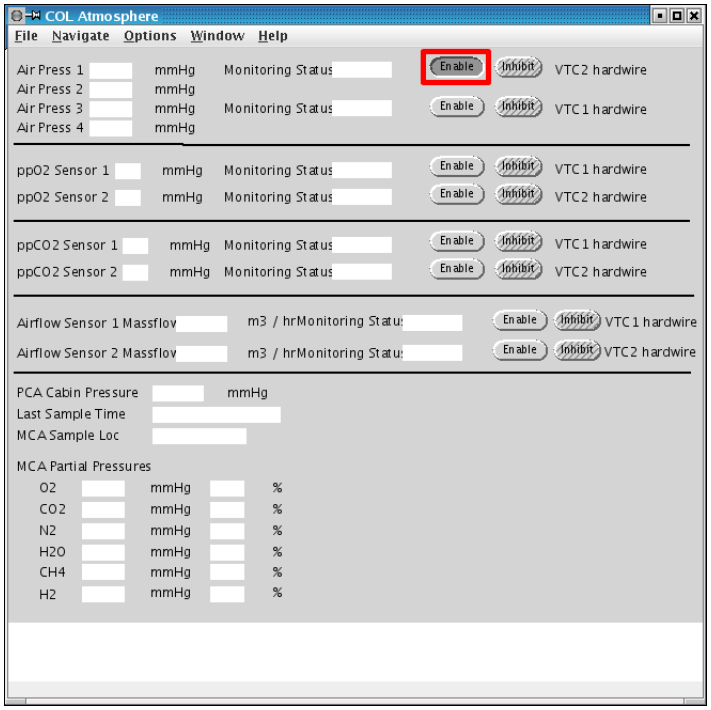
5.4.6.2.1 1-Step-Commanding

To issue command via 1-Step-Commanding:

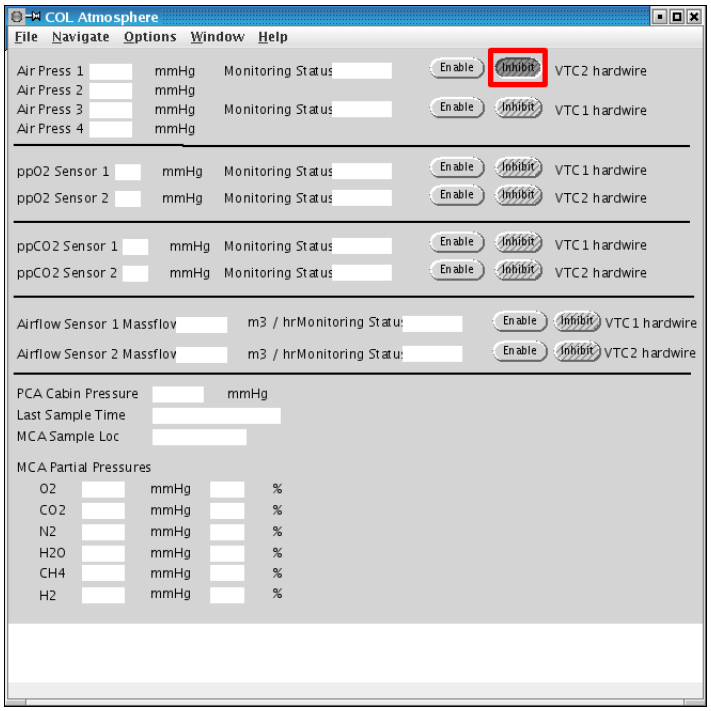
1. Click on command button.

2. Command is sent using one of these methods:

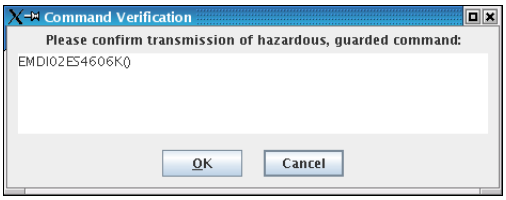
   - Command is asynchronously sent to target system. The command button is immediately available for further command initiation. A sample display for this kind of command is the PCS display **COL Atmosphere** (which can be found under `/examples/import/pcs/xml-all/col_atmosphere.xml` or `/examples/import/pcs/xml/col_atmosphere.xml`).


COL Atmosphere
File Navigate Options Window Help
Air Press 1 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
Air Press 2 mmHg
Air Press 3 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
Air Press 4 mmHg
ppO2 Sensor 1 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
ppO2 Sensor 2 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
ppCO2 Sensor 1 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
ppCO2 Sensor 2 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
Airflow Sensor 1 Massflov m3 / hrMonitoring Statu Enable Inhibit VTC1 hardwire
Airflow Sensor 2 Massflov m3 / hrMonitoring Statu Enable Inhibit VTC2 hardwire
PCA Cabin Pressure mmHg
Last Sample Time
MCA Sample Loc
MCA Partial Pressures
O2 mmHg %
CO2 mmHg %
N2 mmHg %
H2O mmHg %
CH4 mmHg %
H2 mmHg %

- Command must be confirmed. Commands that have to be confirmed are guarded commands and can be recognized by buttons which have diagonal stripes.


COL Atmosphere
File Navigate Options Window Help
Air Press 1 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
Air Press 2 mmHg
Air Press 3 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
Air Press 4 mmHg
ppO2 Sensor 1 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
ppO2 Sensor 2 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
ppCO2 Sensor 1 mmHg Monitoring Status Enable Inhibit VTC1 hardwire
ppCO2 Sensor 2 mmHg Monitoring Status Enable Inhibit VTC2 hardwire
Airflow Sensor 1 Massflov m3 / hrMonitoring Statu Enable Inhibit VTC1 hardwire
Airflow Sensor 2 Massflov m3 / hrMonitoring Statu Enable Inhibit VTC2 hardwire
PCA Cabin Pressure mmHg
Last Sample Time
MCA Sample Loc
MCA Partial Pressures
O2 mmHg %
CO2 mmHg %
N2 mmHg %
H2O mmHg %
CH4 mmHg %
H2 mmHg %

Executor shows confirmation dialog and sends command to target system if user confirms.



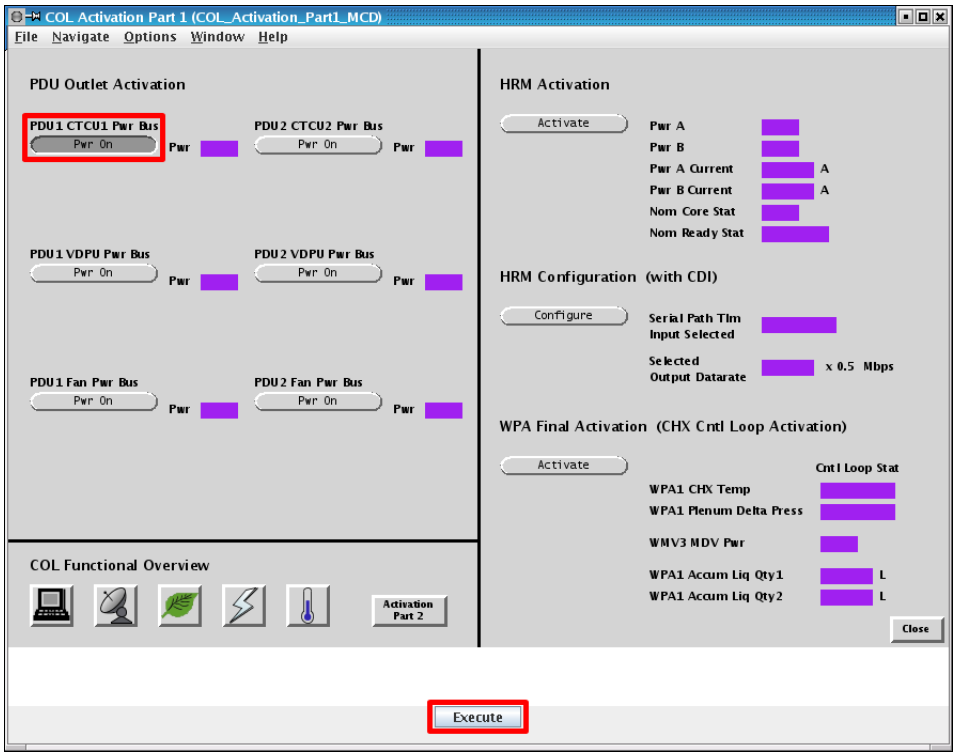
3. The command processing state (issued, accepted/rejected, executed/aborted, success/failure) is shown in the command response window.

5.4.6.2.2 2-Step-Commanding

To issue command via 2-Step-Commanding:

1. Click **command button** (an example display for this is **COL Activation Part 1** which can be found under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ACTIVATION/ACT_PT_1.uss`). This button remains pressed. All other command buttons are released. The execute button is changed to enabled state. If you made a mistake and don't want that command to be executed click another command button. Previously selected button is deselected and the new one is selected (radio-button behavior).

2. Click **Execute**. The command associated with the selected button is sent to target system. Selected button is deselected and execute button becomes disabled again.



5.4.6.3 See also

- Issuing Telecommand via Command List

5.4.7 Issuing Telecommand via Command List

5.4.7.1 Prerequisites

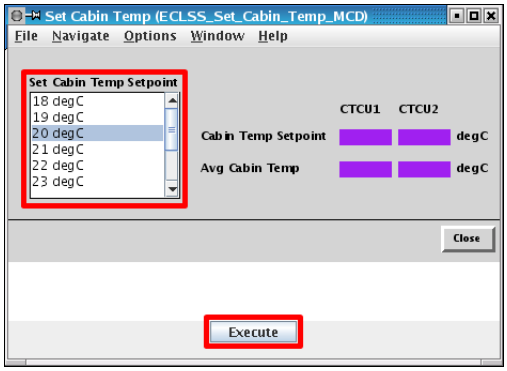
- Selected display contains command list.

5.4.7.2 Issuing Telecommand via Command List

You can issue telecommands by selecting a command from a list and pressing the execute button. The scrollable command lists allow you to select telecommands or UCL commands out of popup menus during run-time execution. As an example you can find a command list in the display **Set Cabin Temp** which can be found under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ECLSS/CABIN_TEMP.uss`).

To issue telecommand via command list:

1. Select command from list.



2. Click **Execute**.

3. Command is processed as described in Issuing telecommand via command button section 2-Step-Commanding.

5.4.8 Finding Displays with Parameter References

5.4.8.1 Prerequisites

- Project root folder must be set and existing. Project root folder must contain displays.

5.4.8.2 Finding Display with Parameter References

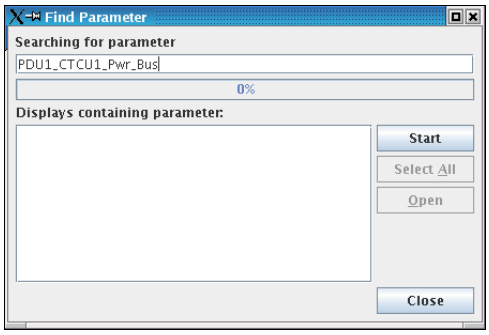
You can find and optionally open displays where opsnames, SIDs, PUIs or pathnames bound to the selected element are also used. The search is performed over the set of displays contained in the project root folder and its sub-folders.

5.4.8.2.1 Find Parameter with No Display Open

1. Choose **Navigate > Find Parameter...** .

2. In the popup window enter the parameter you like to search for in the text field labeled **Searching for parameter**.

3. Click **Start**.



4. If that wasn't the parameter you intended to search for you can stop the search by clicking **Stop** and entering a new opsname, SID, PUI or pathname to search for into the text field below the label **Searching for parameter**. Then click **Start**.

5. Select display from list.

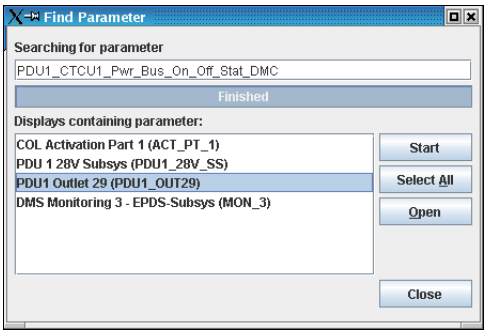
6. Click **Open**.

7. Click **Close**.

Executor opens selected display.

5.4.8.2.2 Find Parameter from Open Display

1. Right-click on any element with parameter binding.
2. Choose **Find Parameter in Other Displays** from pop-up context menu.
3. A window pops up with a list of all displays containing the selected parameter.
4. The search is automatically started with that parameter. If that wasn't the parameter you intended to search for you can stop the search by clicking **Stop** and entering a new opsname, SID, PUI or pathname to search for into the text field below the label "Searching for parameter". Then click **Start**.
5. Select display from list. Or click **Select All**.



6. Click **Open**.
7. Click **Close**.

Executor opens selected display.

5.4.9 Finding Text in Display

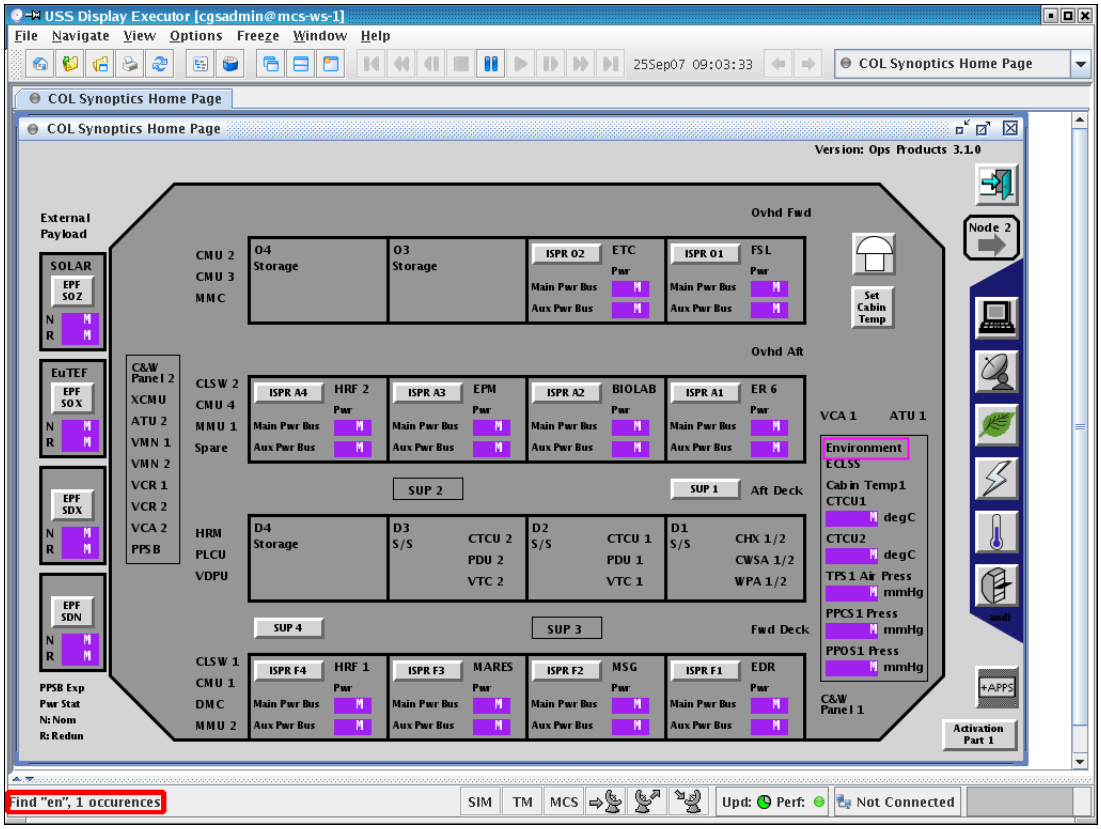
5.4.9.1 Prerequisites

- At least one display must be open and selected.

5.4.9.2 Finding Text in Display

You can find text in displays. This is done by using incremental search.
To find text in display:

1. Choose **Navigate > Find Text As You Type**.
2. In the status-bar the text **"Starting - find as you type"** is displayed.
3. Enter text. Each time a new character is typed, it is appended to the previously entered text. All elements where the entered text appears (either as static text or as a parameter name bound to that element) are highlighted, which means that the text is surrounded by a magenta box. Status-bar shows how many occurrences have been found (in this case 1 occurrence for typing "en").



4. Press **Esc** to finish. Highlighting is removed. Status-bar shows **Find stopped**.

5.4.10 Showing Parameter Values in a Quick Graph

The executor provides a quick graph that can be interactively configured to show values over time (see Section 5.5 for details).

5.4.10.1 Prerequisites

- Move the mouse over a parameter element, e.g., a data field.

5.4.10.2 Showing Parameter Values in a Quick Graph

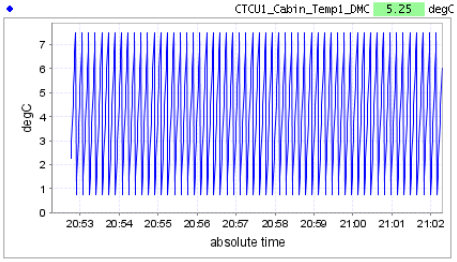
USS can display history data of a parameter in a quick graph.

To show a quick graph for a parameter shown in a data field do the following:

1. Right-click on parameter element (e.g. **Cabin Temp1** in display **Cabin Temperature Control Unit 1**, which can be found under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ECLSS/CTCU1.uss`).

2. Select **Quick Graph** from context menu.

3. A non-modal window with a line graph for the selected parameter opens.



4. Click **Close** to close the graph window or leave it open as long as you need it.

The line graph in the quick graph window automatically updates when the parameter shown receives an update. Note that no curve is drawn when the parameter status is invalid (not aquired, static, etc.). This also means that parts of a curve can be missing when a parameter was switching from nominal to missing and nominal again. If you don't see a curve where you would expect one, always check the parameter details first (see Section 5.4.2.1). One possible reason for a value not being shown is when a parameter is acquired via playback with an 'old' acquisition time that is older than what the quick graph buffer can hold.

The time span to show in the graph can be configured via executor preferences. See Section 5.9.1.9 for details on how to do this.

The quick graph can combine multiple parameters in a strip chart (same domain axis for all parameters). Just repeat the procedure above and all further parameters will be added to the already open quick graph.

The contents of a quick graph can be saved, printed or exported as HTML report just like a normal display. These functions are available via the File menu entries in the main menu of the quick graph window.

5.5 Quick Graph

5.5.1 Introduction

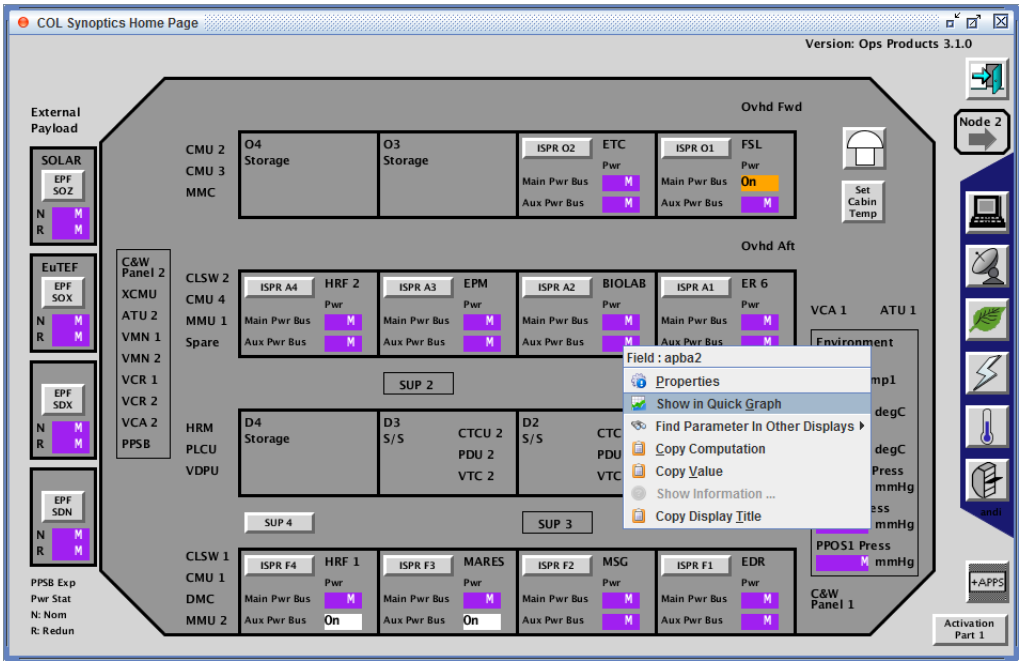
The executor provides a quick graph that can be interactively configured to show parameter values over time. The quick graph shows a strip graph providing a single horizontal time-based axis and multiple (vertically stacked) sub-graphs each having a dedicated vertical value-based axis. The quick graph is a display window that behaves similar to synoptic displays and can be docked, closed, frozen, etc. The contents of a quick graph can be saved, printed or exported as HTML report just like a normal display. These functions are available via the File menu entries in the main menu when the quick graph window has the focus. The curves of the sub-graphs (drawn in their plot-areas) are automatically updated when the parameters receive new updates. Note that no curve is drawn for a parameter with invalid status like "not acquired" or static. This also means that parts of a curve can be missing when a parameter is switching from nominal to missing and back again. If there is no curve drawn even if a curve is expected, the parameter details should be opened to check the current status. A quick graph has a legend which is placed above the plot. The legend has two goals: (1) it uniquely associates a curve in the plot with a parameter; (2) it shows the current value and state of a parameter.

5.5.2 Adding and Removing Parameters

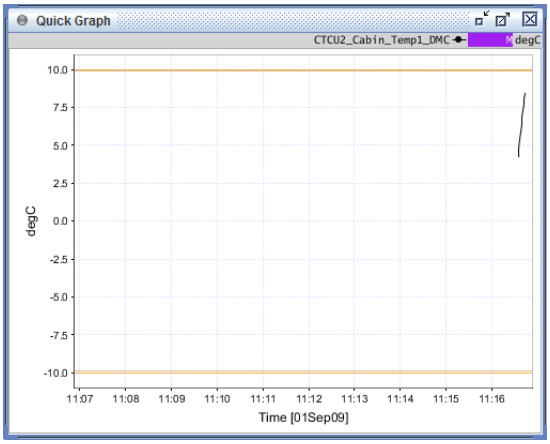
A new quick graph can be created by selecting the "Show in Quick Graph" entry that appears in the context menu of all field elements. Showing a parameter in a quick graph means either that a new quick graph display will be created or that an existing quick graph display will be brought to the front and a new sub-plot is added. This interactive procedure can be repeated to create a quick graph having multiple sub-plots that are stacked one upon the other (new sub-plots will always be added to the bottom). Each sub-plot (except for the last) can be removed from the quick graph by selecting the "Remove Plot" entry in the context menu that appears on performing a right mouse click into the appropriate sub-plot area.

To show a quick graph with two sub-plots do the following:

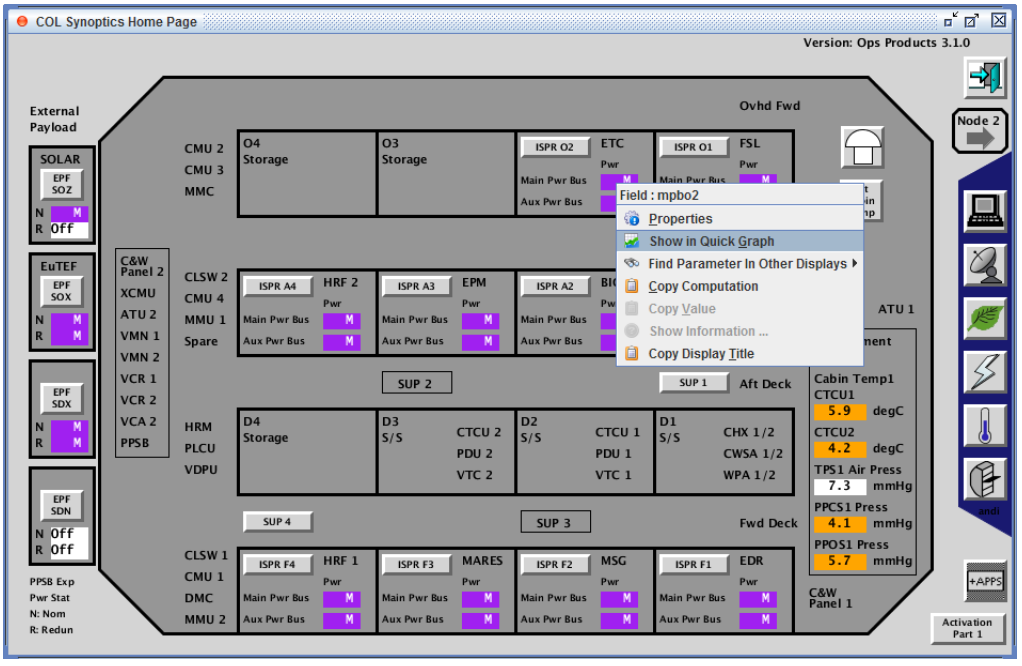
1. Right-click on a field element and select **Show in Quick Graph** from the context menu.



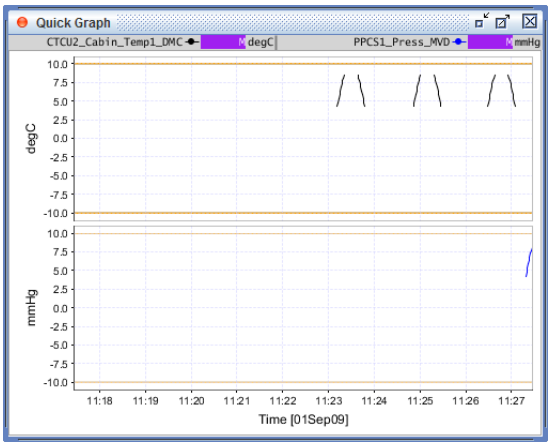
2. The initial quick graph is created as a new display. The quick graph has one sub-plot showing the curve for the selected parameter.



3. Right-click on a second field element and select **Show in Quick Graph** from the context menu.

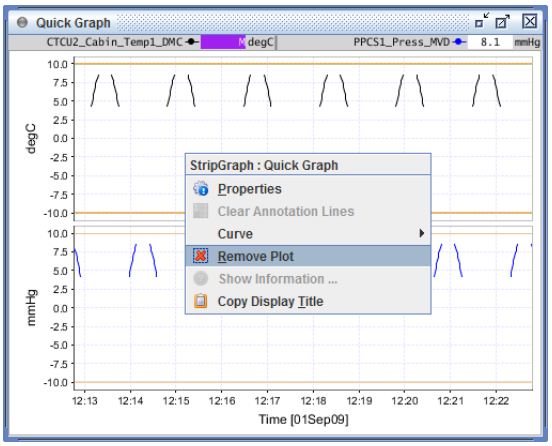


4. A new sub-plot is added to the bottom of the existing quick graph.

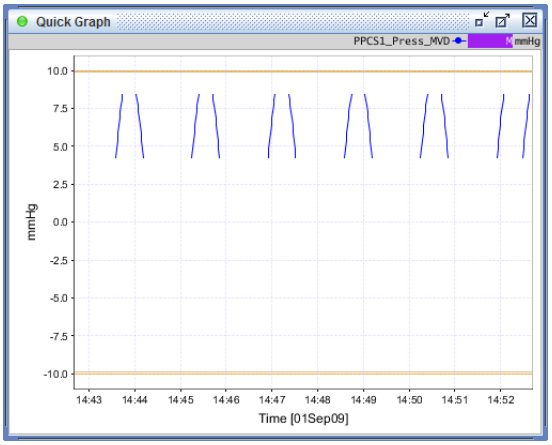


To remove a sub-plot from an existing quick graph (having multiple sub-plots) do the following:

1. Right-click in a certain sub-plot area of an existing quick graph and select **Remove Plot** from the context menu.



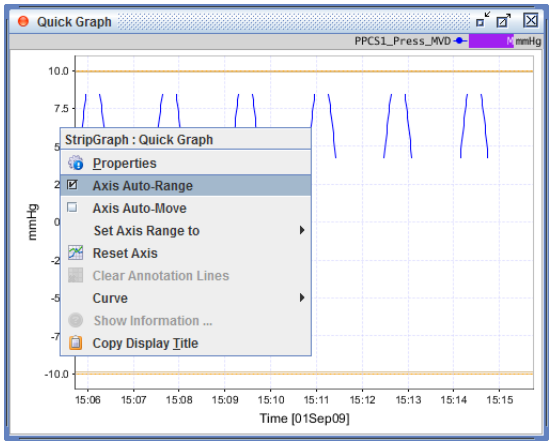
2. The chosen sub-plot is removed from the quick graph.



5.5.3 Configuring the Axis Modes

A quick graph axis can be configured to always show a static (non-moving) range or it can be automatically updated depending on the incoming data values by two different axis modes: (1) Auto-range: an axis in auto-range mode does dynamically adjust the axis range to always include the minimum and maximum of all existing data values that are maintained in the memory buffer; (2) Auto-move: an axis in auto-move mode uses a fixed range that is dynamically moved to always show the recently received data value at the right-most (or top) axis position. Note that the auto-move mode is applied only if auto-range mode is disabled and that a static (non-moving) range is applied only if both modes are disabled.

The axis mode can be configured in the context menu that appears on performing a right click on an axis:

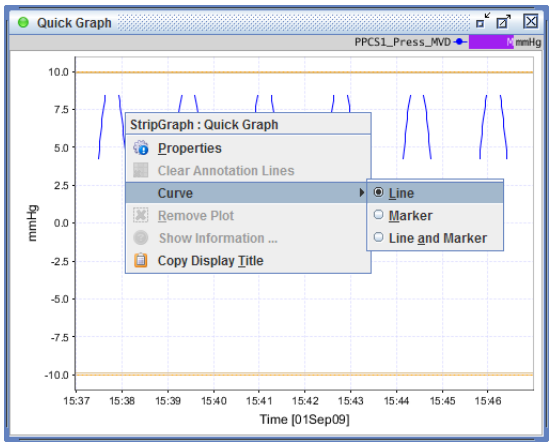


5.5.4 Switching the Curve Style

Points of nominal data values on a curve in a quick graph can be emphasized by value markers. Adjacent points of nominal data values are connected either by straight lines (linear interpolation) or by step lines (no interpolation). Non-nominal data values are not connected to adjacent nominal points by lines and are not emphasized by value markers. Thus a single non-nominal data value in the middle of two nominal data values results in an interruption of the curve.

Note that in the worst case of alternating nominal and non-nominal data values no curve is drawn at all. Thus it is highly recommended to always enable value markers to see single nominal data values placed between non-nominal data values.

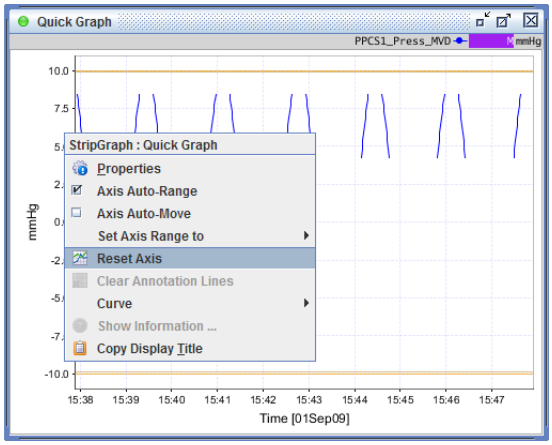
The curve style can be configured in the context menu that appears on performing a right click on the quick graph display:



5.5.5 Scrolling and Zooming an Axis

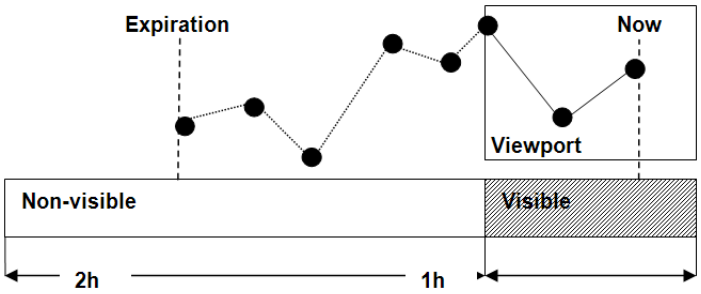
A quick graph axis can be interactively scrolled and zoomed by using mouse interactions. To manually scroll an axis, the mouse cursor has to be moved over the axis and the left mouse button has to be clicked and held before dragging the mouse in the appropriate direction. To manually zoom an axis, the mouse cursor has to be moved over the axis and the left mouse button has to be double-clicked and held before dragging the mouse in the appropriate direction. This causes an instant axis rescaling around the click position, whereas a dragging to the up or right direction causes a zoom-out and a dragging to the down or left direction causes a zoom-in.

Note that clicking an axis immediately deactivates auto-range and auto-move modes. Any quick graph axis can be resetted to its defaults by double clicking the axis with the left mouse button or by selecting the proper context menu entry.



5.5.6 Understanding the Buffer and Viewport

A quick graph keeps the data values for a specific duration in a dedicated memory buffer. The duration is affected by the expiration modes (to be configured in the executor preferences) and is fully independent of the visible area of a sub-plot. This visible area as defined by the rectangle that is spanned by the horizontal and vertical axis ranges is called "viewport". The viewport can be thought of as a window that allows viewing a specific part of the available data in the memory buffer. Each viewport can be freely moved or resized (by scrolling and zooming the axes) so that all available data becomes visible.

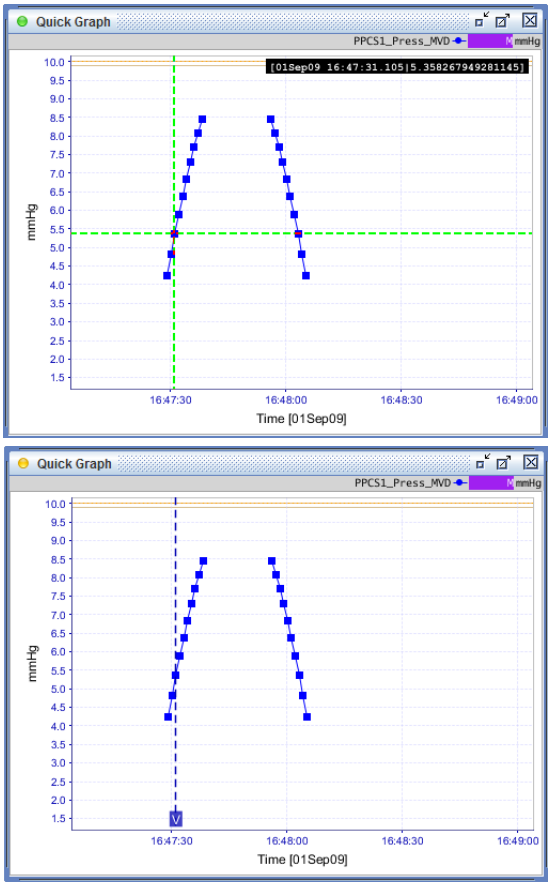


5.5.7 Managing Time and Value Annotations

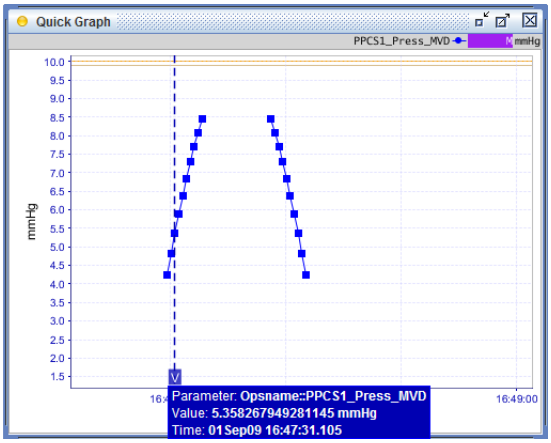
A quick graph supports vertical annotation lines that can be manually added and removed to ease the process of visual curve analysis. Annotation lines can be managed by using a cross hair that appears when holding the CTRL-key and moving the mouse cursor over the plot. In this mode two different types of annotation lines can be added to a quick graph by just clicking with the left mouse button into a certain sub-plot area: (1) placing the mouse cursor on top of an actual data value causes the cross hair to become green and clicking with the left mouse button creates a new value annotation; (2) placing the mouse cursor in an empty sub-plot area and clicking with the left mouse button creates a new time annotation. All annotations are composed of a vertical line and a small text box showing a single character. Value annotations are marked with a 'V' and time annotations with a 'T' character. Both character boxes can be clicked to get additional information in a floating text box. Annotations can be removed by holding the CTRL-key, placing the mouse cursor on top of the abbreviation character (the cross hair becomes red) and clicking with the left mouse button.

Note that the cross hair can be hidden by just leaving the plot area when holding the CTRL-key at the same time.

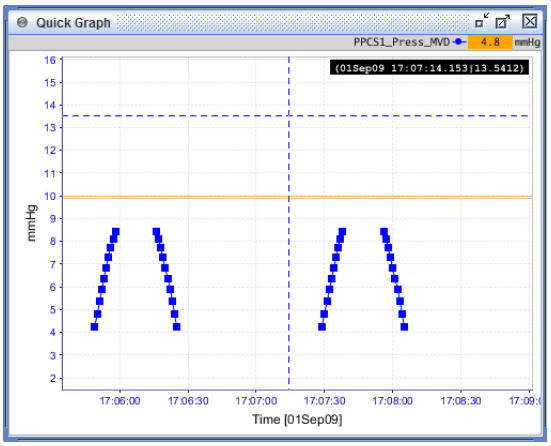
A value annotation can be added by performing a left mouse click on a certain data value (the cross hair is drawn green):

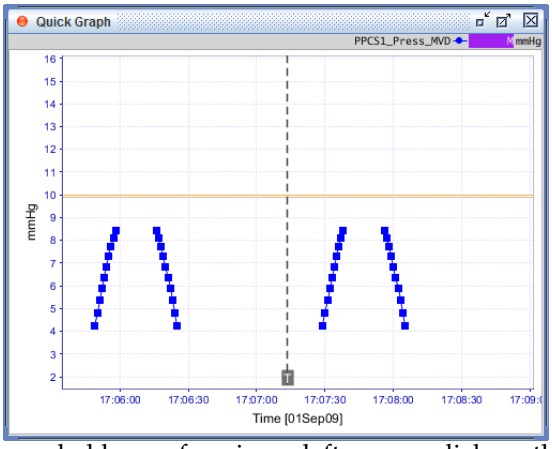


A value annotation can be expanded by performing a left mouse click on the character abbreviation box:

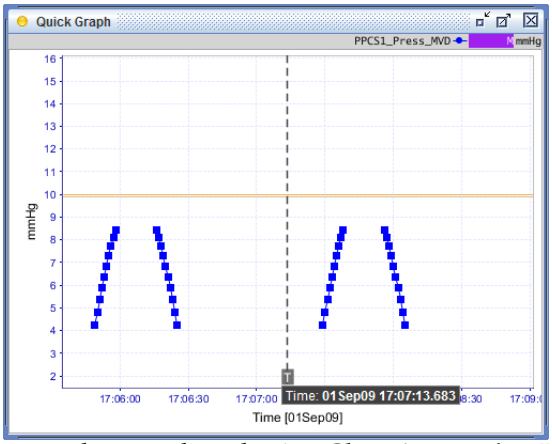


A time annotation can be added by performing a left mouse click into an empty area (the cross hair is drawn blue):

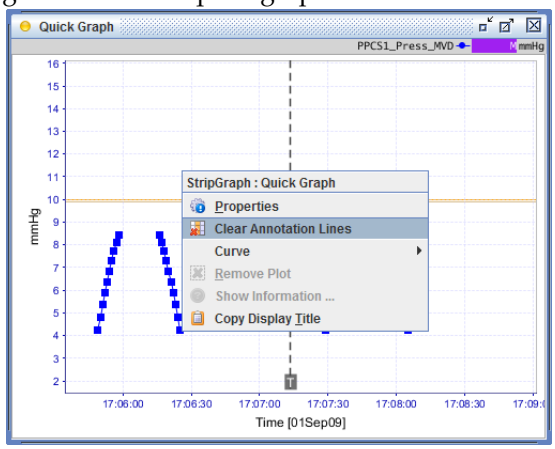




A time annotation can be expanded by performing a left mouse click on the character abbreviation box:



All annotation lines can be removed at once by selecting **Clear Annotation Lines** in the context menu that appears on performing a right click on the quick graph:

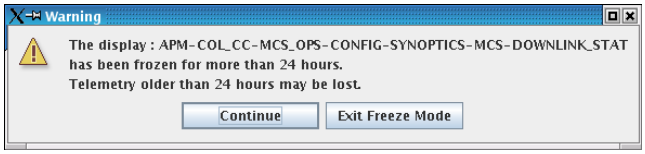


5.6 Freeze Mode

5.6.1 Introduction

The default mode for a display is the real-time mode in which the display immediately displays telemetry received from the connected target system. All real-time telemetry received for a display is stored in a disk based repository for up to 24 hours. When a display is set into freeze mode the display of real-time telemetry is suspended making it possible to navigate through the previously recorded telemetry for this display via video recorder like controls. The earliest point in time that can be displayed during freeze mode is the moment when the frozen display has been opened in the executor. The maximum freeze mode duration can be configured in the freeze mode preferences.

When a display is kept in freeze mode for longer than the configured duration, the following warning dialog will be displayed, because now the oldest data might get lost.


Warning
The display : APM-COL_CC-MCS_OPS-CONFIG-SYNOPTICS-MCS-DOWNLINK_STAT
has been frozen for more than 24 hours.
Telemetry older than 24 hours may be lost.
Continue
Exit Freeze Mode

The dialog gives 2 options:

1. **Continue :** Displays stays in freeze mode and the user accepts that the oldest data for the display might get lost.
2. **Exit Freeze Mode :** Displays returns to real-time.

5.6.2 The Freeze Mode Controls

To use the freeze mode in the executor it has to be enabled first in the freeze mode preferences. This will make the freeze mode available for all displays opened after change of the setting.

The following table lists the freeze mode control buttons and their effect on a display.

Table 5.4: Freeze mode controls

Button	Operation
Goto Beginning.	The display time is set to the point in time when the frozen display was opened.
Fast Backward.	Performs a major step backwards in time. The default duration for a major step is 60 seconds. The value for major step duration can be changed in the freeze mode preferences.
Step Backward.	Performs a step back in time to the previous change of single or multiple telemetry items on the display. Note that this operation will skip periods of time for which no telemetry is available in a single step.
Stop (Exit Freeze Mode).	Exits the freeze mode and restores the latest telemetry received for the frozen display.
Pause (Enter Freeze Mode).	Suspends real-time updates for a single display or stops a freeze mode play operation. No telemetry for a frozen display is lost if the display is frozen for no longer than 24 hours.
Play.	Starts a playback of telemetry at the current display time. The playback can be stopped with the pause operation.
Step Forward.	Performs a step forward in time to the next change of single or multiple telemetry items on the display. Note that this operation will skip periods of time for which no telemetry is available in a single step.
Fast Forward.	Performs a major step forward in time. The default duration for a major step is 60 seconds. The value for major step duration can be changed in the freeze mode preferences.
Goto End.	The display time is set to the latest point in time for which telemetry has been received for the display.

5.7 Help

5.7.1 Showing Display Help

5.7.1.1 Prerequisites

- Help file must exist. URL prefix must have been configured for executor.

5.7.1.2 Showing Display Help

You can view the HTML help of the display which has been provided by the display's author.

To display help:

1. Choose **Help > For Display: name of display** menu command. If no help file has been defined for the display, the menu item is disabled.

2. Executor shows HTML help for selected display in web browser.

5.7.2 Getting Executor Version Information

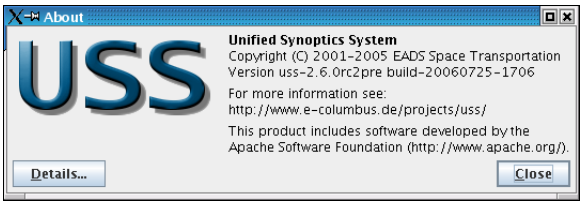
5.7.2.1 Prerequisites

- Executor must be running.

5.7.2.2 Getting Executor Version Information

To get the executor version information:

- Choose **Help > About...** .
- Executor shows version information in popup dialog.



5.8 Miscellaneous

5.8.1 Print Preview

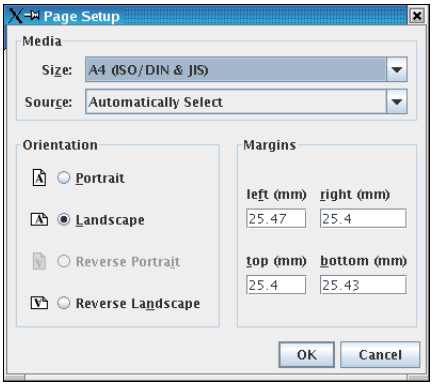
5.8.1.1 Prerequisites

- Display must be selected.

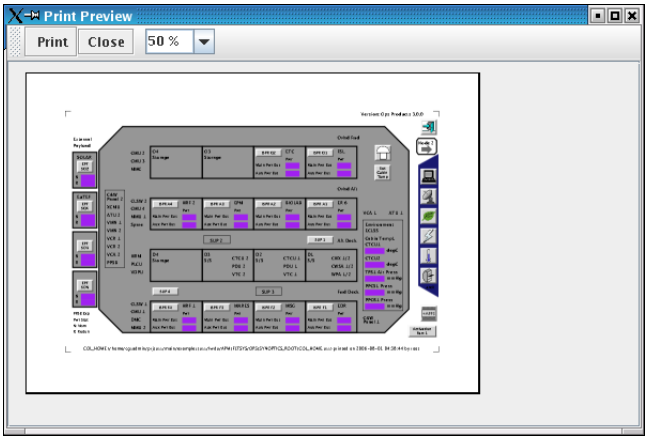
5.8.1.2 Print Preview

To preview print:

1. Choose **File > Print Preview...** .
2. In the popup dialog change any printer settings if necessary.



3. Click **Ok**.
4. In the print preview popup select percentage to change the zoom.



5. Click **Print**.

6. Click **Close**.

Executor prints image of display.

5.8.2 Printing Display

5.8.2.1 Prerequisites

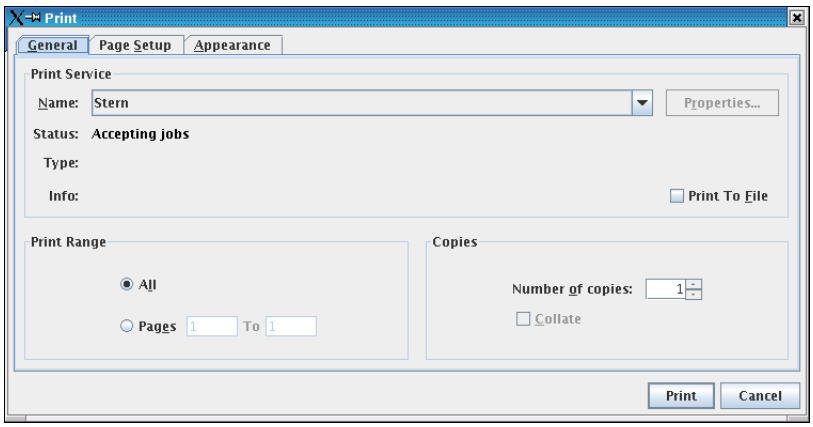
- Display must be selected.

5.8.2.2 Printing Display

USS allows you to print displays.
To print display:

1. Choose **File > Print...** .

2. Select printer and optionally configure print setup.



3. Click **Print**.

Executor prints image of display. It is not possible to print parts of the display. Only the whole display can be printed.

5.8.3 Creating Display Snapshot

5.8.3.1 Prerequisites

- Display must be selected.

5.8.3.2 Creating Display Snapshot

You can save snapshots of a running display as graphics file for later use. This feature is useful for presentations, documentations and reports which automatically are complemented with display screen-shots.

To create display snapshot:

1. Choose **File > Save Snapshot**.

Executor creates and automatically saves image into configured directory (usually under /home/user/.uss-x.y.z). The file name consist of the display name plus an appended time stamp. Snapshot is saved as PNG.

5.8.4 Saving a Copy of Current Display

5.8.4.1 Prerequisites

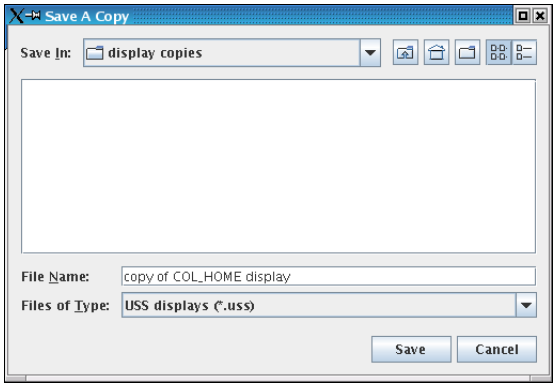
- Display must be selected.

5.8.4.2 Saving a Copy of Current Display

You can save the current display to a file for further processing for example editing.

To save a copy of current display:

1. Choose **File > Save A Copy...** .
2. In the file chooser dialog select folder in which you want to save your copy.
3. Enter file name under which you wish to save the copy.



4. Click **Save**.

Display is saved in USS XML format.

5.8.5 Configuring Status Display

5.8.5.1 Prerequisites

- USS must be installed.

5.8.5.2 Configuring Status Display

The MCS status display is displayed in the status-bar at the bottom of the executor window. It can be turned off as follows.

To configure status display:

1. Open **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor.
2. To not display the status display change the value of **uss.mcs.status.indicator** property to false.

```
### Enable the MCS status indicator in the executor status area.

uss.mcs.status.indicator = false
```

The MCS status display will not be shown in USS executor area at the bottom of the window in the status-bar.

5.8.5.3 See also

- Section 1.2

5.8.6 Configuring Data Quality Indicators

5.8.6.1 Prerequisites

- USS must be installed.

5.8.6.2 Configuring Data Quality Indicators

You can configure the visual appearance of data quality indicators (DQI).

To configure data quality indicators:

1. Edit **uss.properties** file in **$basedir/uss-x.y.z/etc/** with a text editor and set the corresponding property to the name of a DQI configuration file.
2.
```
### Different Data Quality Indicator (DQI) styles are chosen dependent
### on which target system the display is designed for. Each pair of
### entries defines the mapping file for one target system.

uss.view.dqistyle.file = ${basedir}/etc/pws_dqistyle.xml

uss.view.dqistyle.target1 = PWS
uss.view.dqistyle.file1 = ${basedir}/etc/pws_dqistyle.xml

uss.view.dqistyle.target2 = PCS
uss.view.dqistyle.file2 = ${basedir}/etc/pcs_dqistyle.xml

uss.view.dqistyle.target3 = MCS
uss.view.dqistyle.file3 = ${basedir}/etc/mcs_dqistyle.xml
```

Executor uses this file for rendering DQIs.

5.8.6.3 See also

- Section 1.2

5.9 Configuring the Executor

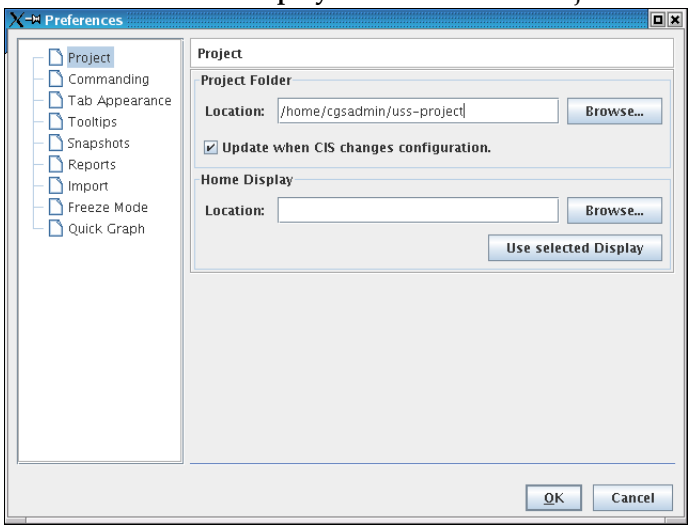
In this section we'll tackle all available preferences which allow the user to **configure the executor**. For each preference we'll fully explain the corresponding dialog.

5.9.1 Preferences

To configure the preferences choose **Options > Preferences...** . Following preferences are available: **Project, Commanding, Tab Appearance, Tooltips, Snapshots, Reports, Import, Freeze Mode and Quick Graph**. The corresponding dialogs and the options therein are explained in the next subsections.

5.9.1.1 Project

Preferences for the **Project Folder** and **Home Display** can be set under Project.



5.9.1.1.1 Setting Project Folder

- Click **Browse...** button.
- In the popup dialog select folder you like to use.
- Click **Choose Directory**.

Check or uncheck **Update when CIS changes configuration** depending on whether the folder shall be updated when the configuration has been changed on CIS.

5.9.1.1.2 Setting Home Display

To use currently selected display as home display:

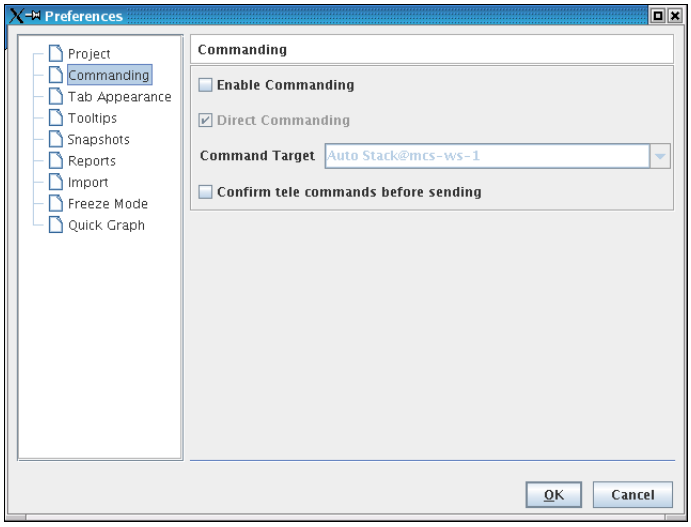
- Click either **Use selected Display** to set currently selected display as home display.

To browse for a display to set as home display:

- Click **Browse...** button.
- In the popup dialog select display you like to set as home display.
- Click **Choose USS Home Display.**

5.9.1.2 Commanding

Preferences for commanding include **enabling of commands**, enabling of **direct commanding**, choosing the **command target** and **confirmation of commands**.



5.9.1.2.1 Enabling Commanding

- Check **Enable Commanding** if you want to use commanding.

5.9.1.2.2 Enabling Direct Commanding

- Check **Direct Commanding** if you want to use direct commanding.

5.9.1.2.3 Choosing Command Target

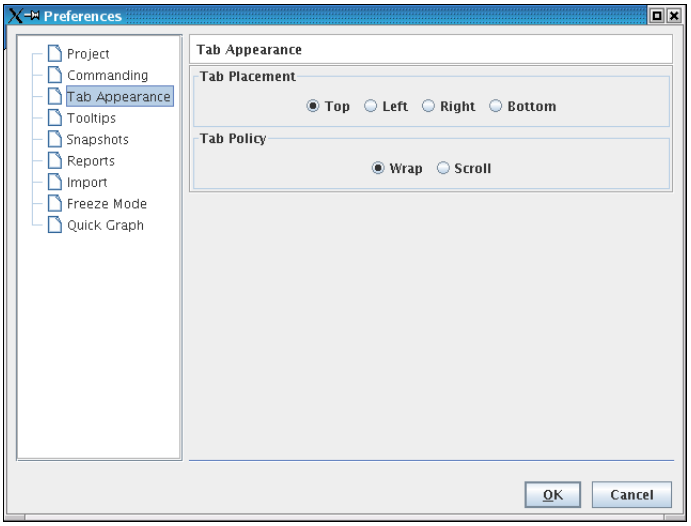
- In the drop down list next to the label **Command Target** select the command target you like to use.

5.9.1.2.4 Confirming Tele Commands

- Check **Confirm tele commands before sending** when you wish the tele command to be confirmed after sending.

5.9.1.3 Tab Appearance

Preferences for the tab appearance include both the **Tab Placement** as well as the **Tab Policy**.



5.9.1.3.1 Setting Tab Placement

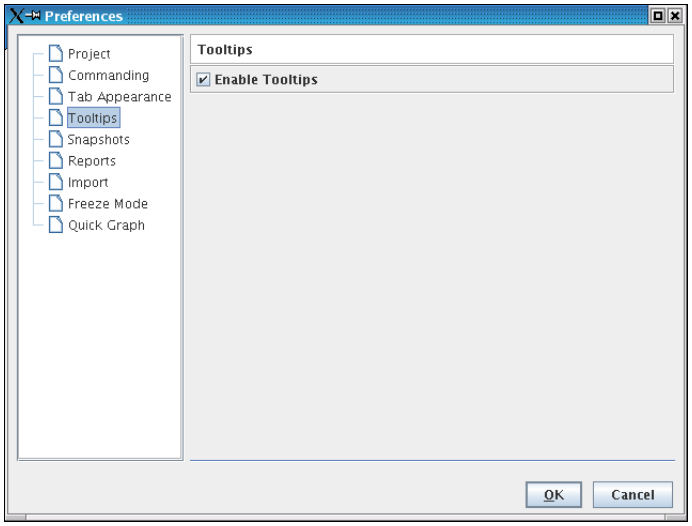
- Click one of the radio buttons (**Top, Left, Right, Bottom**). The tabs will be placed at the chosen position.

5.9.1.3.2 Setting Tab Policy

- Click either **Wrap** or **Scroll**.

5.9.1.4 Tooltips

Preferences for tooltips can be set to either show the tooltips or not.

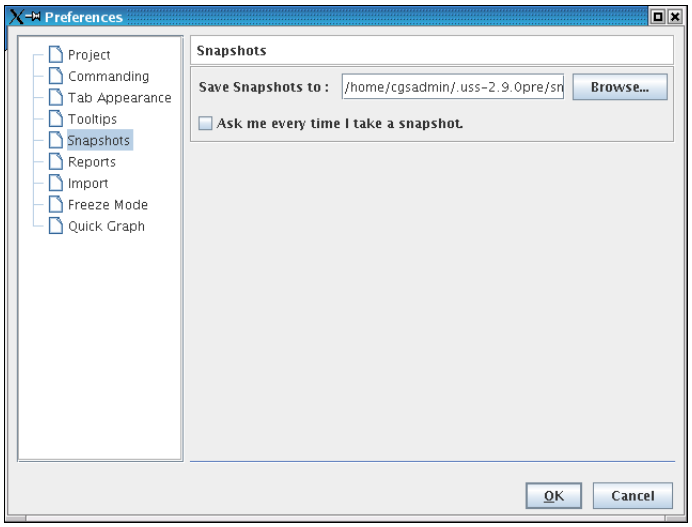


5.9.1.4.1 Enabling tooltips

- Check **Enable Tooltips** if you want the tooltips to be shown.

5.9.1.5 Snapshots

For snapshot creation the default folder where the snapshots will be saved can be set.



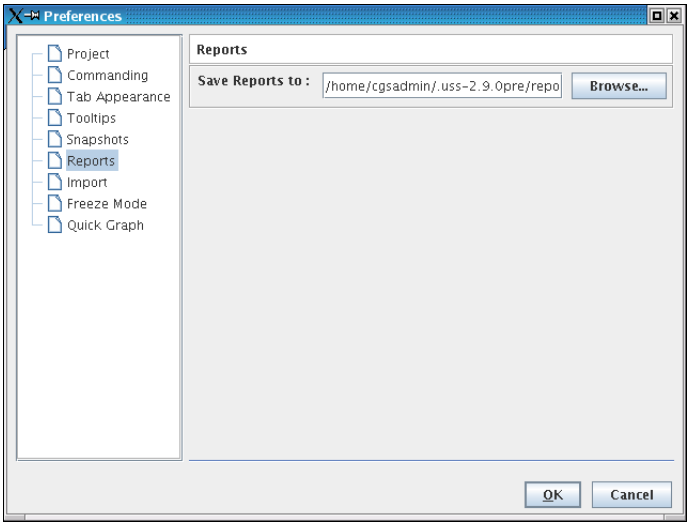
5.9.1.5.1 Saving Snapshots

- Click **Browse...** button.
- In the popup dialog select folder in which you like to save the snapshots.
- Click **Choose Directory**.

Check **Ask me every time I take a snapshot** when you want to select the folder every time you take a snapshot.

5.9.1.6 Reports

Generated reports are saved to the folder which is set in the preferences.

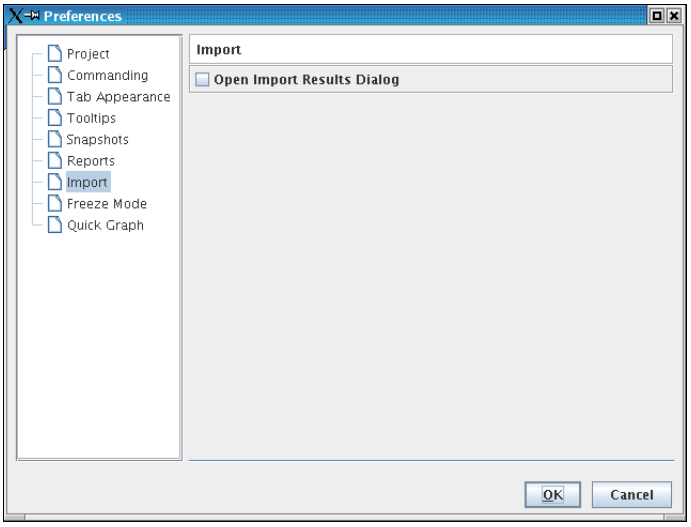


5.9.1.6.1 Setting Report Folder

- Click **Browse...** button.
- In the popup dialog select the folder in which you like the reports to be saved.
- Click **Choose Directory**.

5.9.1.7 Import

While importing a display the import results are being logged in case an error is encountered. You can set the preference to open a dialog showing the import results when an error was encountered.

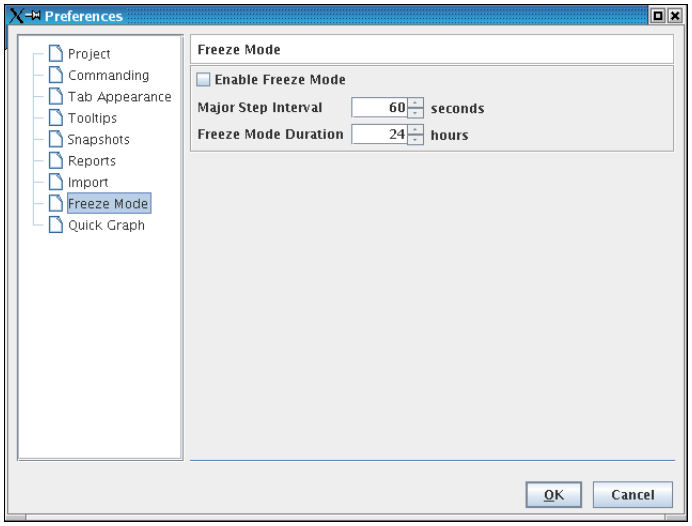


5.9.1.7.1 Opening Import Results Dialog

- Check **Open Import Results Dialog** when you want a popup to be displayed that shows the import results which have been encountered while importing a display.

5.9.1.8 Freeze Mode

Preferences for the freeze mode include its enabling, the setting of the **major step interval** and the setting of the **freeze mode duration**.



5.9.1.8.1 Enabling Freeze Mode

- Check **Enable Freeze Mode** to enable freeze mode. Note that this setting will not be applied to displays already open in the executor.

5.9.1.8.2 Setting Major Step Interval

- Click the spinner next to **Major Step Interval** to either increase or decrease the interval. The setting defines the interval skipped by a fast forward or fast backward freeze mode operation.

5.9.1.8.3 Setting Freeze Mode Duration

- Click the spinner next to **Freeze Mode Duration** to either increase or decrease the duration. The setting defines the maximum duration a display can be kept in freeze mode without data loss.

  Note that a long Freeze Mode Duration might consume a lot of disk space on the drive where the users home directory is located. The default value is 24 hours which can result in a 1 GB large database.

5.9.1.9 Quick Graph

Preferences for quick graph includes defining method of expiration for data values/samples. Expiration of data values/samples is controlled via **time period** and **number of samples**. To specify **no limit** enter 0. **Note:** All data values displayed in quick graph are stored in memory. Setting one or both limits to 0 may cause the application to run out of memory. This may happen if period is longer than 24 hours or if number of samples is higher than 10,000 or if too many displays are executing.

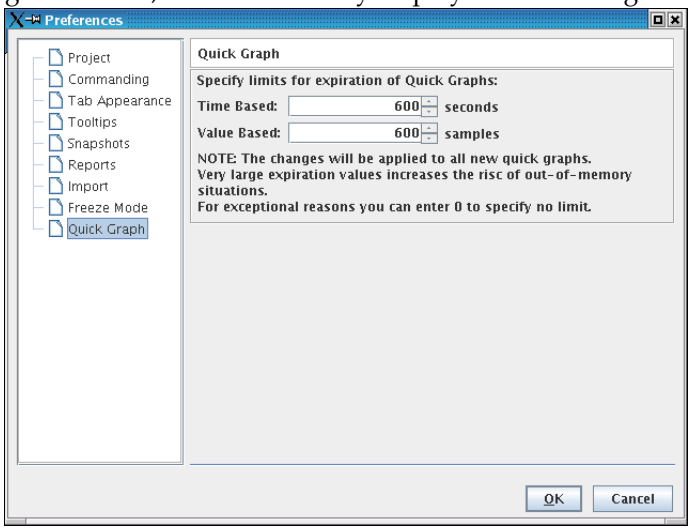


Table 5.5: Setting limits

Time Based	Value Based	Consequence
0	0	Application will run out of memory.
60,000	0	Application will run out of memory, if samples are very big.
0	6000	Application will run out of memory, if frequency of samples is too high.
60,000	6000	Application will run out of memory, when first limit is exceeded.

5.9.1.9.1 Set time based limit

- Click spinner next to **Time Based** to either increase or decrease the time period. Or enter value into field.

5.9.1.9.2 Set value based limit

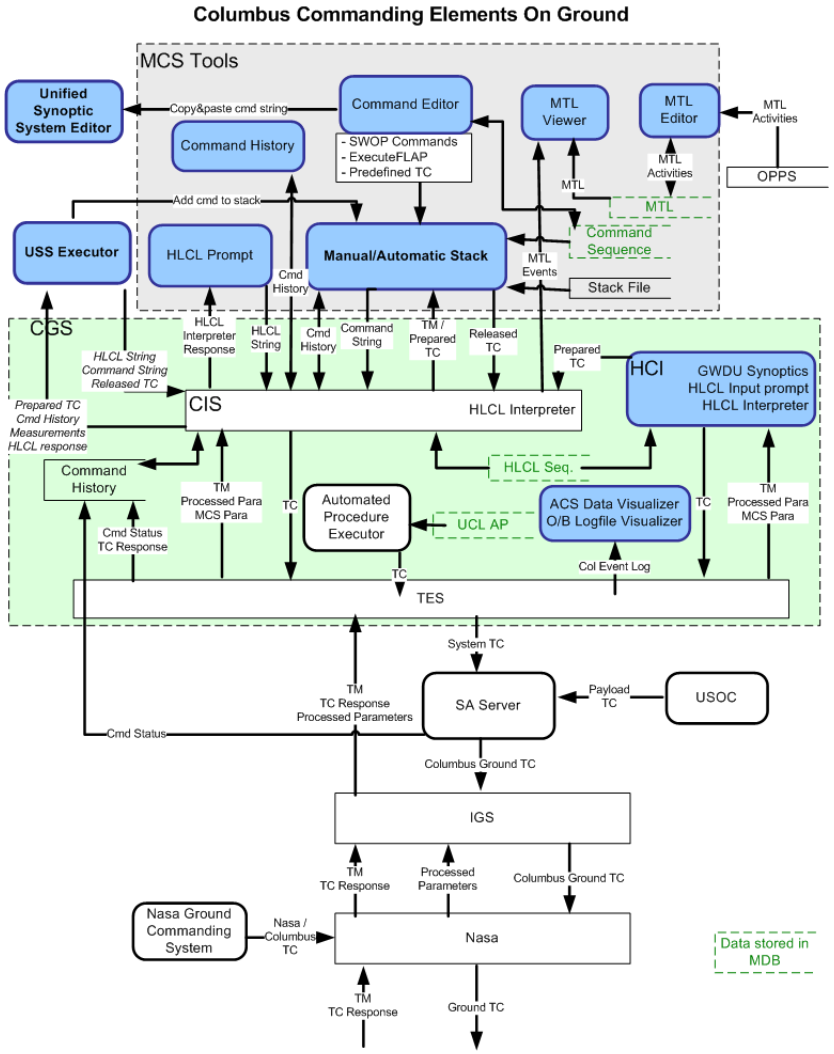
- Click spinner next to **Value Based** to either increase or decrease the total number of samples. Or enter value into field.

Chapter 6

Application Scenarios

6.1 MCS

Ground telecommanding with USS integrated into MCS is described in this section. An overview of the commanding scenario is given in the following diagram.



- **Edit TC**: A telecommand (SWOP command, executeFLAP, Predefined TC) can be edited in the command editor of MCS Tools. The command string is then copied and pasted into a command button of a display opened in the USS editor.
- **Edit HLCL**: A ground command (HLCL) can only directly be edited in the USS editor because the MCS command editor supports only telecommands.

- **Edit structured command**: All commands can be edited as structured commands using the structured command editor in USS. A structured command is not edited as a command string but as nested structure of commands and their single parameters. For each parameter dedicated properties can be defined like value ranges, types and default values. For structured command input fields can be associated to certain parameters of the command. An input field in the synoptic display which is related to a command parameter enables the user to specify the command parameter value at runtime.

- **Configure command target**: Before execution of a telecommand in a flight synoptic display from ground the user has to configure the command execution target. This can be configured in the preferences window of the USS executor (refer Switching targets for commands). In the section "Commanding" the following options can be selected:

  - Commanding can be enabled/disabled, i.e. it has to be enabled if the user wants to issue a command.
  - Direct commanding can be enabled which means that the command is sent directly to the CGS CIS.
  - Direct commanding can be disabled which means that the command is added to a command stack within MCS Tools.
  - The command target specifies to which command stack the command shall be added, either the manual stack or the auto stack. In the manual stack the command is queued and the user interactively has to enable and then send the command. In the auto stack the command is queued and sent automatically as far as the auto mode is activated.

- **Configure command stack**: The target command stack has to be enabled as remote command receiver in MCS Tools. This is specified in the preferences window of MCS Tools. In the menu item Options > Preferences > Remote Command Execution > Remote command receiver the appropriate command stack can be selected (manual stack or auto stack).

- **Direct TC commanding**: When pushing a command button associated to a telecommand in the USS executor the command string is sent to CIS for command preparation. The CIS returns the prepared command as CCSDS packet. After successful preparation the command is released via CIS. The command is transmitted to the onboard system. After execution onboard the response packet is sent back to ground. The command results are displayed in the command status window of the USS executor. The command results can also be displayed in the command history window of MCS Tools.

- **TC commanding via command stack**: When pushing a command button associated to a telecommand in the USS executor the command is added to the command stack of MCS Tools. The command stack sends the command string to CIS for command preparation. The CIS returns the prepared command as CCSDS packet and the command stack queues the command. In the manual stack the user has to enable the command and sent it to be released via CIS. In the auto stack running in auto mode this is performed automatically. The command is transmitted to the onboard system. After execution onboard the response packet is sent back to ground. The command results are displayed in the command history window of MCS Tools.

- **HLCL commanding**: When pushing a command button associated to a ground command (HLCL) in the USS executor the command string is sent to CIS for execution via HLCL interpreter. The HLCL interpreter results are displayed in the command status window of the USS executor.

Current commanding limitations in USS

- Command pre/post conditions defined in the MDB are evaluated from CGS and the result is displayed in the command response window. Furthermore the result of the post conditions is shown in the MCS Tools command history window as TMV status. If the command is sent via command stack, the pre-conditions are evaluated prior to command release, i.e. a command is not sent if the pre-condition is not fulfilled.

- Dedicated pre/post conditions for command instances cannot be defined in a synoptic display. This could be a future enhancement if required.

- Hazardous commands sent in direct commanding mode are checked via CGS and rejected because no password has been specified. The command executor displays the CGS error and prompts the user to enter the password.

Hints how to edit onboard commands in USS
For more details see section Command Buttons.

- Attaching a command to a command button should be performed via drag and drop mechanism selecting a command from a list of available commands. This way the command definition copied to the display is inline with the definition in the SCOE data. For each mandatory parameter having no default value an input field will be created. If the parameter has constraints these are also attached to the input field. The user may set a default value for a parameter.
- If for any reason the constraints shall be updated take care to use only a more restrictive definition than was set in the SCOE command definition. In other words an updated constraint should always be a subset of its orign.
- If a non-mandatory parameter shall be explicitely set to a value the user has to create an input field for it, add constraints if applicable, and set a default value. This could be overwritten onboard. Take care to correctly set the constraints being valid against the SCOE definition.
- Executing the consistency checker on a display performs a complete check of all commands. In particular it checks the names of the parameters, their types as well as their constraints if applicable. As a consequence the consistency checker must be executed on a display before installing it onboard. When being executed onboard the commands can be updated only via the input fields. If a value is written into an input field the constraint mechanism automatically checks it.
- The execution of the USS consistency checker can be started in the USS editor for the actual edited display. For a set of displays the consistency checker is started as command line call:

```
uss-consistency-checker.[sh|bat] [OPTION(S)]... ccu_int_version display_dir
```

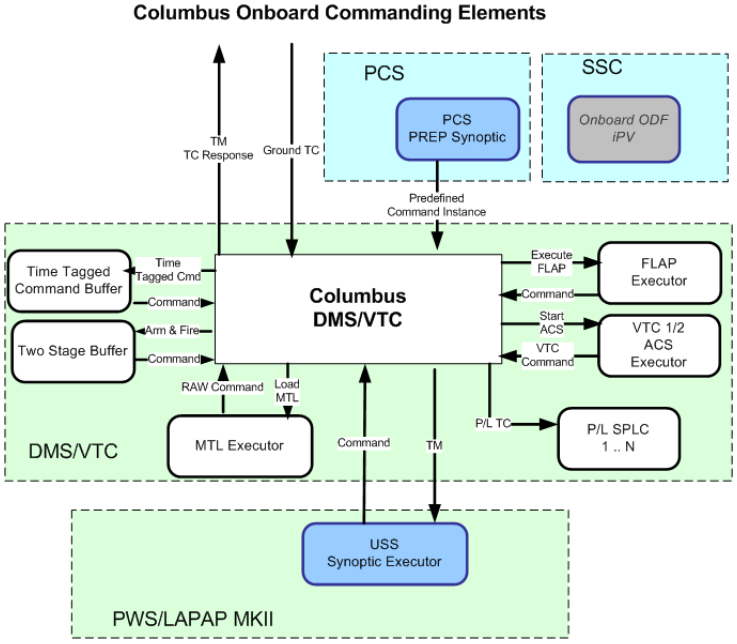
`OPTION`: Specify `-h` for getting a complete list of available options. Default option setting is: print report as plain text to standard output, search for SCOE data in current directory, and set test configuration to zero.

`ccuVersion`: CCU internal version number of CCU against which the displays shall be checked.

`display_dir`: OS path where to find the display(s) to be checked.

6.2 LAPAP MKII

Columbus onboard commanding with USS integrated into LAPAP MKII is described in this section. An overview of the commanding scenario is given in the following diagram.



Automated onboard command checks in LAPAP MKII

During and after ground display development the display consistency checker verifies the onboard commands. Onboard only the input fields attached to command parameters can be used to update the command. The input fields are connected to constraints. An automated check mechanism will verify input values typed into the input field. Therefore an onboard command is always verified before execution.

The only exception from this rule are parameters representing end item references. These can not be checked onboard because there is no end item information available. The check is performed by the DMS. The DMS rejects invalid commands.

Current commanding limitations in LAPAP MKII

- Only structured commands (replacement of CPL) can be used.
- No HLCL commanding is supported.
- The system command checker is not supported.
- Only DMS commands supported, no VTC commands.
- Only direct commanding supported.

6.3 COL-TRE

The application in COL-TRE is the execution of command instances defined in the Flat File I/F which are referenced via NASA Command PUIs. For further details refer to COL-TRE documentation.

Chapter 7

Importing Foreign Display Formats

USS provides support for importing several non-USS display formats. This chapter describes how foreign displays can be converted to USS displays so that they subsequently can be edited and executed with USS. In many cases, displays do not need to be converted in an extra step, but can be imported "on-the-fly" in the editor and executor. However, for large sets of displays, a batch import may be more convenient and efficient.

USS provides support for importing the following display formats:

- PCS/PREP: These are NASA displays in XML format designed for execution on the PCS laptop.
- PWS/FWDU: These are ESA Columbus displays in SAMMI format designed for execution on the PWS laptop.
- GWDU: These are displays in Dataviews/Gipsy format designed for execution on ground facilities.
- Satmon: These are displays in Satmon format used, for example, in the Columbus Control Center.

There are three different ways to use these foreign display formats in USS:

1. Directly opening a single display with the "File Open" dialog in the editor (see Section 4.3) or executor (see Section 5.3.3.2). The display can be immediately used and optionally saved in USS format. This is analogous to how you would open and work with a non-Word document in Word.
2. Batch conversion of a whole set of displays. The batch conversion is run from within the editor (see Section 3.4.2.2). The result is saved as a set of displays in USS format.
3. Batch conversion of a whole set of displays. The batch conversion is run as a console application (without GUI). The result is saved as a set of displays in USS format. This is the way to go if the conversion shall be integrated in another larger batch process without requiring user interaction. Details are described in the following sections.

The conversion process for each format is described in more detail in the following sections. However, here are some general guidelines that apply to all conversions:

- If converting extracted displays from another system (e.g. GWDU) into USS format use new empty temporary directories for conversion output. Do not do any manual conversions under the `uss-project` directory.
- If adding new USS displays from an external source (e.g. GWDU conversion output) to MDB, first copy all files to the `uss-project` directory. Be careful to use the right directory names and file names (both must map to valid MDB path and end-item names).

7.1 PCS/PREP Displays

This section describes how NASA PCS/PREP displays are imported and converted to USS format. PCS displays are XML files with the extension `.xml`. PCS displays can be directly loaded in the editor and executor. USS needs two more files containing additional information about the displays. These files are

called `command_instances.csv` and `pui-mapping.csv`. The first one contains information about command PUIs in the display and is used to determine the criticality of a command. The PUI mapping file is used to retrieve the Opsname for a parameter PUI. Both files are searched in the same directory as the display or in any of its parent directories. See the example displays that are delivered with USS for an example of how the configuration files should look like.

Example 7.1 Batch-converting a set of PCS displays

The following command reads all PCS displays in the directory `examples/import/pcs/xml`, converts them to USS format, and writes the result into the output directory `/tmp`.

```
uss-convert.sh -o /tmp examples/import/pcs/xml
```

7.2 PWS/FWDU Displays

This section describes how PWS/FWDU displays are imported and converted to USS format.

The import of FWDU displays involves four steps:

1. Extracting the FWDU displays from MDB.
2. Generate XML SCOE files containing all referenced data if not yet available.
3. Converting the Sammi ASCII format to USS format.

General Guidelines:

- If converting extracted displays from FWDU format into USS format use new empty temporary directories for conversion output. Do not do any manual conversions under the uss-project directory.
- If Adding new .uss displays from external source (e.g. FWDU conversion output) to MDB, first copy files to uss-project directory. Be careful to use the right directory names and file names (both must map to valid MDB path and end item names).

7.2.1 Extracting FWDU Displays from MDB

The first step to extract FWDU displays is to read the FWDU ASCII definition of each display from the MDB.

Prerequisites are:

- MCS / CGS installation with MDB, and MDA at a minimum.
- CCU version containing the FWDU synoptic display(s) to be extracted and all end items referenced by the display(s).

To extract the displays from CGS/MDB proceed as follows:

```
uss-fwdu-extract.[sh|bat] [OPTION]... -c ccuVersion targetDir
```

- `OPTION`: Specify `-h` for getting a complete list of available options.

  Default extract is to process all displays of the specified CCU. To extract only one display use the -e option.

  Default MDB connection is this one of the Unix user currently logged in. No extra options are necessary.

  To connect to another MDB instance or to start the FWDU extract from an MS Windows PC use the according options.

- `ccuVersion`: CCU internal version number of CCU where displays shall be retrieved from.
- `targetDir`: Directory where to store the finalized FWDU display files, relative to USS installation directory or absolute path.

For example, if you want to extract data from APM_MASTER fill in ccuVersion and targetDir:

```
uss-fwdu-extract.[sh|bat] -i 149.243.225.45 -o 1521 -s p5oracle -u genguest -c ↩
    ccuVersion targetDir
```

7.2.2 Generating SCOE XML files

All in the FWDU synoptic displays referenced data must be available for offline access. These data are organized in the so called XML SCOE files. To generate XML SCOE files issue the following command:

```
$MCS_HOME/mcs_tools/bin/common/generate_scoexml.sh CCU ccu-version 0
```

- `ccu-version`: CCU internal version number. You will get this number using the IMDB CU version manager on the CCU version.

The generated files will be stored in `$MCS_HOME/mcs_tools/data/scoeXML/<ccu-version>_0_*.xml`. Copy or move these files to directory `<USSinstallationdir>/share/scoe`

7.2.3 Converting the FWDU ASCII definition to USS Format

The final step is to convert the directory tree containing the `.fwdu` files to `.uss` files.

```
uss-convert.[sh|bat] [OPTION]... -o dest-dir source-dir
```

- `OPTION`: Specify `-h` for getting a complete list of available options.
- `dest-dir`: Directory where to store the FWDU display files converted to USS with extension .uss
- `source-dir`: Directory containing FWDU display files with extension .fwdu.

7.2.4 Create HTML report of original PWS/FWDU displays

This section describes how a HTML report of original PWS/FWDU displays can be created in two steps.

1. As described above (Section 7.2.3) convert PWS/FWDU displays to USS format using option `-e` wich will disable the enhanced conversion of command lists and images.

   ```
   uss-convert.[sh|bat] -e -o dest-dir source-dir
   ```

2. Create HTML report of converted displays. Now source directory is destination directory of first step.

   ```
   uss-report.[sh|bat] -o dest-dir source-dir
   ```

7.3 GWDU Displays

The import of GWDU displays involves the following steps:

1. Extracting the GWDU displays from MDB and converting from binary Dataviews format to XML report format.
2. Generate XML SCOE files containing all referenced data if not yet available.
3. Converting the XML report to USS format including creation of a symbol library if applicable.
4. Store USS displays in MDB as new end items.

These steps are available as an integrated process in CGS. For details please refer to the CGS User Manual, section 'USS Displays'.

7.3.1 GWDU to USS Conversion Details

This section describes in detail how GWDU display elements are mapped to USS elements.

7.3.1.1 Conversion of GWDU Display Objects

The table given below lists for each GWDU display object the target USS object. In case of special handling a comment explains the details.

Table 7.1: GWDU display object conversion to USS object

GWDU Object	USS Object	Comment
Static Objects		
Line	Polyline	With two points
Arc	Arc	
Rectangle	Polyline	Closed polyline
Ellipse	Ellipse	
Circle	Ellipse	
Polyline	Polyline	
Text	Label	
Vector text	Label	
Scalable fonts text	Label	In case of setting width and height the best fit font size will be calculated. No scaling of fonts.
Subdrawing	Dynamic Symbol	The GWDU Symbols used for the sub-drawing must be re-edited as SVG or PNG graphics and a mapping table from GWDU symbol to USS symbol needs to be provided. The symbols will be included in a user symbol library.
Symbol	Symbol	Static symbol The GWDU Symbols used must be re-edited as SVG or PNG graphics and a mapping table from GWDU symbol to USS symbol needs to be provided. The symbols will be included in a user symbol library.
Icon	Image	Will be transferred as reference to a PNG file.
Image	Image	Will be transferred as reference to a PNG file.
Inputs		
Button	Command button	Command buttons should not be used to open displays via HLCL because command buttons are not IDAGS compatible for navigation. Use Picture select button instead.
Popup menu	*Popup menu or Command List*	Command List is fixed, not popup
Picture select	Navigation button	
Text menu	Command list	
Dynamic objects		
Bar graph	Bar graph	Vertical, only one sample
Bar line graph	Bar graph	Vertical, only one sample
Packed bar graph	Bar graph	Vertical, only one sample
Packed bar line graph	Bar graph	Vertical, only one sample
Stacked packed bar line graph	Strip chart line graph	
Center bar graph	Bar graph	Vertical, only one sample
Horizontal bar graph	Bar graph	Horizontal, only one sample
Pigback bar graph	Bar graph	Vertical, only one sample
Pickback dist bar graph	Bar graph	Vertical, only one sample

Table 7.1: (continued)

GWDU Object	USS Object	Comment
Solid bar graph	Bar graph	Vertical, only one sample
Step graph	Line graph	With attribute 'step'
High low graph	--	
High low bar graph	--	
High low line graph	Line graph	
Line graph	Line graph	
Stacked line graph	Strip chart line graph	
Filled line graph	Line graph	
Filled line stacked graph	Strip chart line graph	
Filled line dist graph	Line graph	
Strip chart graph	Line graph	
Raster strip chart graph	Line graph	
Stacked strip chart graph	Strip chart line graph	
Vertical strip chart graph	Line graph	Vertical
Raster vertical strip chart graph	Line graph	vertical
Waterfall graph	--	
Raster waterfall graph	--	
Point chart graph	--	
Point line graph	Line graph	With value markers
Spectro graph	--	
Stacked spectro graph	--	
Smooth spectro graph	--	
Smooth stacked spectro graph	--	
Pie chart graph	--	
Radial graph	--	
Radial NE graph	--	
Realtime line graph	Line graph	With 'realtime' attribute
Realtime step graph	Line graph	With 'realtime' and 'step' attributes
Scatter graph	--	
Impulse graph	--	
Impulse to zero graph	--	
Web graph	--	
Multi-Y web graph	--	
Surface graph	--	
Vector graph	--	
Flowfield graph	--	
Contour graph	--	
Filled contour graph	--	
Blocks graph	--	
Packed blocks graph	--	
Indicator graph	--	
Controller graph	--	
Horizontal controller graph	--	
Fader graph	Bar meter	
Dials graph	Elliptic meter	
Dials with history graph	Elliptic meter	
Dial 360 graph	Elliptic meter	

Table 7.1: (continued)

GWDU Object	USS Object	Comment
Meter graph	Elliptic meter	
Knob graph	Elliptic meter	
Fan graph	Elliptic meter	
Analog clock graph	Elliptic meter	
Artificial horizon graph	--	
Bullseye graph	--	
Face graph	--	
Dynamic drawing graph	--	
Moving drawing graph	--	
Digits graph	Data field	
Text graph	Data field	
Message graph	Data field	
Legend graph	--	
Color graph	--	
Size graph	--	
Box graph	--	
Circle graph	--	
Triangle graph	--	

7.3.1.2 Conversion of GWDU Attributes

GWDU attributes are the formatting details of GWDU display objects.

The following table lists for all GWDU attributes the way of conversion into USS properties. In case of special handling a comment explains the details.

Table 7.2: GWDU attributes conversion to USS properties

GWDU Attribute	USS Property	Comment
Text font	Lucida sans, Lucida, Lucida typewriter	Will be translated in a 'best fit manner'
Text size	Font size or bounding box	Depending on source definition
Text format (bold, italic, underline, color)	Text format (bold, italic, underline, color)	
Text direction	--	Only horizontal
Text rotation	Steps of 90 degrees allowed	
Line type	Solid or dashed	All non solid types are converted to dashed
Line width	1 pt .. 4 pt line	
Line color	Line color	
Filling	Fill color and line color	Depending on usage of 'edge' in fill status
Context features of graphs		
Title	Title	Standard format
Samples	Number of samples to be shown	Sample based graph in opposite to time based graph
Format string	Format of data fields	C-like format string, to be implemented in USS V1
Opaque	--	Opaque is default
Grid	--	
Color of bar or curve	--	Default colors

Table 7.2: (continued)

GWDU Attribute	USS Property	Comment
Legend	Default legend	On or off
Time/Value axis label	Domain/Range axis label	Standard format
Ticks	Ticks	On or off
Tick label	Tick label	On or off, standard format
Dynamic behaviour of texts and shapes		
Dynamics of shapes	Color, size, position can be dynamic	
Dynamics of texts	Text is converted to data field	In case of '$OPSNAME' usage dynamics are not converted
Dynamics of subdrawing	Dynamic Symbol	Symbol shown changes according to threshold

7.4 Satmon Displays

NOTE

The import of Satmon displays is a technology demonstrator and has prototype status in the current version of USS. Some testing has been done with example displays, but the Satmon import is not formally validated or qualified.

This section describes how Satmon displays are imported and converted to USS format. Satmon displays are XML files with the extension `.xml`. Satmon displays can be directly loaded in the editor and executor or batch-converted as described in the beginning of this chapter. It is recommended to perform the batch conversion with configuration upate described below before using the displays in the executor.

Example 7.2 Batch-converting a set of Satmon displays

The batch conversion of a set of satmon displays is a 2-step process described in the following example. The following command reads all Satmon displays in the directory `examples/import/satmon`, converts them to USS format, and writes the result into the output directory `/tmp`.

```
uss-convert.sh -o /tmp examples/import/satmon
```

To make the displays usable with a specific configuration the displays have to updated with an xml scoe file for this configuration.
The following command updates the uss displays in directory `/tmp` with configuration information from an xml scoe file with the ccu internal version `316989992`.

```
uss-update-references.sh 316989992 /tmp
```

(see also Section 7.5)

7.4.1 Satmon display file format analysis

This section contains an analysis of the Satmon XML format with respect to compatibility to the concepts of USS. As input for this analysis a set of Satmon FCT display files dated June 2006 and a few screenshots were used. No information about run-time behaviour of the displays was available. As result of this analysis a Satmon importer for USS was implemented.

7.4.1.1 Satmon display XML structure

A Satmon XML display always starts with the "PageDef" root element defining the type, title and dimensions of a display. The "PageDef "root element can contain "FixedAlpha", "OutputAlpha", "PlotDef" and "LeanProcEntry" elements defining the displays contents. In displays of the PageType "custom" only "FixedAlpha" and "OutputAlpha" elements are used. A display of type "lineplot" contains only "PlotDef " elements and displays of the type "leanproc" contains "LeanProcEntry" elements only.

The following table list the attributes of the Satmon "PageDef" root element and their interpretation.

Table 7.3: Satmon "PageDef" root element attributes

PageDef attribute	**Type**	**Value Interpretation**
PageType	String	"custom"= alpha numeric display containing only FixedAlpha and OutputAlpha elements "lineplot" = plot display with one or more PlotDef elements "leanproc" = procedure display containing only LeanProcEntry elements
Title	String	The display title
Width	Integer	The width of the display in number of characters. Character size is assumed as 10 pixels wide and 18 pixels high. This is optional for "lineplot" and "leanproc" displays. Default lineplot display width derived from screenshots is 800. Size for leanproc displays is defined by its contents.
Height	Integer	The height of the display in number of characters. Character size is assumed as 10 pixels wide and 18 pixels high.This is optional for "lineplot" and "leanproc" displays. The lineplot display height is derived from number of PlotDef elements and the plots height. Default plot height is for a display containing multiple plots is 180 pixels. Size for leanproc displays is defined by its contents.

7.4.1.2 Mapping of Satmon display Elements to USS elements

The following table lists known Satmon XML elements and the corresponding USS element.

Table 7.4: Satmon to USS element mapping

Satmon Element	**USS Element**	**Interpretation**
PageDef	Display	Display background color is black. Display target is "MCS" defining a DQI style suitable for displays with black background.
FixedAlpha	Label	Defines a static text. Using monospaced font "Lucida Sans Typewriter"
OutputAlpha	Field	Defines a telemetry field. Using monospaced font "Lucida Sans Typewriter"
PlotDef	Graph	Defines a graph element.
LeanProcEntry	Label	Defines a procedure step

7.4.1.3 Alphanumeric Satmon Displays

This section describes the interpretation of elements used in alphanumeric displays.

Satmon uses style ids in "StyleID" attributes of the text elements "FixedAlpha" and "OutputAlpha" defining the appearance of these text elements. These styles are not distributed with display files, but seem to be part of a Satmon executor installation. The following table lists the ids found in the display set and there interpretation. If an unknown style id is found then the color white is used.

Table 7.5: Satmon text styles for USS version 2.12

Satmon style	Interpretation
NormalFA	Left aligned, color yellow
YellowFA	Left aligned, color yellow
CyanFA	Left aligned, color cyan
Mnemonic	Left aligned, color green

Table 7.6: Satmon text styles added with USS version 2.15.

Satmon style	Interpretation
DkGreyFA	Left aligned, color dark grey
GreyFA	Left aligned, color grey
OrangeFA	Left aligned, color orange
PurpleFA	Left aligned, color purple
WhiteFA	Left aligned, color white
NormalOA	Left aligned, color white
RightOA	Left aligned, color white
UninitOA	Left aligned, color yellow
CyanUL	Left aligned, color cyan
YellowUL	Left aligned, color yellow
LPChkBgrd	Left aligned, color black with dark grey background

The following table lists the attributes and their interpretation for a FixedAlpha static text output element.

Table 7.7: FixedAlpha attributes

FixedAlpha attribute	Type	Interpretation
row	Integer	The y-position of the element in number of characters.
col	Integer	The x-position of the element in number of characters.
text	String	The static text to display
StyleID	String	The style to use for text presentation. Defines colour and alignment (see Table 7.5).
Hotlink	String	Optional attribute containing a link to a display. If this is present the text is displayed in italic and the label is generated as NavigationButton for the specified display.

The following table lists the attributes and their interpretation for a OutputAlpha output element for displaying telemetry.

Table 7.8: OutputAlpha attributes

OutputAlpha attribute	Type	Interpretation
row	Integer	The y-position of the element in number of characters.
col	Integer	The x-position of the element in number of characters.
text	String	The static text to display

Table 7.8: (continued)

OutputAlpha attribute	**Type**	**Interpretation**
Param	String	The opsname of the configuration item presented by this OutputAlpha. If this has no value no visual representation will be created (See display 1005.xml). NOTE If an identifier other than opsname is used, no output will be displayed for this measurement.
StyleID	String	Not used (see Table 7.5). USS fields use a DQI style definition. For Satmon displays the MCS style is used. NOTE The Satmon runtime color coding for acquisition and monitoring states is unknown and will probably differ from the MCS definition.
Length	Integer	The length of this OutputAlpha in number of characters
OutConv	String	The output conversion to perform: "A" mapped to USS type String "F" mapped to USS type Float "I" mapped to USS type Integer USS will always present the engineering value even if these assumptions are wrong. Number formatting will only be performed as defined in "FormatStr" if type Float has been detected for the item. NOTE This type detection is not reliable. As a result field contents might not be formatted correctly and draw outside of their bounds defined by "Length". The type detection could be improved if the CGS type from the configuration was known for every item.
FormatStr	String	The length of this OutputAlpha in number of characters
LADTID	Integer	Unknown (Value is always 6)
ShowTail	Boolean	Unknown (Value is always 1)
Alignment	Integer	Unknown (Value is always 0)

7.4.1.4 Plot Displays

This section describes the interpretation of elements used in plot displays.

Table 7.9: PlotDef attributes

PlotDef attribute	**Type**	**Interpretation**
TimeDuration	Integer	The domain axis duration in seconds
Height	Integer	Optional height in number of characters
YAxisCaption	String	The text for the range axis label
YMin	Float	Range axis min value
YMax	Float	Range axis max value
YAutoScale	Boolean	If this is "true" then auto scaling of the range axis is performed to match value range of received telemetry. This overrides the range set by YMin and YMax.
LabelColour	String	Color string in hex format #BBGGRR
DotsOnly	Boolean	Show value markers only if this is "true" and do not draw a line between values.
TimeLabelFormat	String	Not supported (Seems to be always 'hh:nn') USS automatically formats the time label showing also seconds and milliseconds if possible.
TimeParamMnemonic	String	Not supported (Known values are 'UTC' and 'CSECYEAR'. USS always uses the mission time as time base for graphs)
YTickSep	Integer	Not supported
LargeTickSep	Integer	Not supported
XAxisColour	String	Unknown (always #008000)
YAxisColour	String	Unknown (always #008000)
YAxisLabelFormat	String	Unknown (no usages found)
TimeParamLADTID	Integer	Unknown (always 6)
IgnoreNoData	Boolean	Unknown (always false)

The following table describes the attributes of the "PlotLineDef" element used inside "PlotDef" elements to bind a line in a plot to a parameter and define its appearance.

Table 7.10: PlotLineDef attributes

PlotLineDef attribute	**Type**	**Interpretation**
LineColour	String	Color of the plot line in hex format #BBGGRR
Mnemonic	String	The opsname of the configuration item presented by this plot line. NOTE All datasources are created with type "Float" which is correct for most line plots but will result in no plot being displayed for discrete measurements. This can only be handled correctly when performing the conversion with configuration information.
Staircase	Boolean	If this is set to "true" this line is drawn as a step graph otherwise samples are directly connected with a line. Satmon seems to allow mixing of step and line plots in one graph which is not supported by USS graphs. Found no usage of mixed step / line graphs in existing displays.
Title	String	Not Supported. The title for this line used in the graph legend. USS graphs always display the opsname which also seems to be the Satmon default.

Table 7.10: (continued)

PlotLineDef attribute	Type	Interpretation
Hotlink	String	Not Supported. Found no usage in existing Satmon displays.
LineNum	Integer	Unknown
LADTableID	Integer	Unknown (Value is always 6)
MinGap	Integer	Unknown

7.4.1.5 Procedure Displays

This section describes the interpretation of elements used in procedure displays.

Table 7.11: LeanProcEntry attributes

LeanProcEntry attribute	Type	Interpretation
row	Integer	The x position of the procedure step / step title in number of characters
Param	String	If there is a param this must be a step otherwise this is a step title. Steps are displayed with style YellowFa. Step titles are displayed with style Mnemonic. With USS version 2.15 a telemetry field is displayed for the parameter.
Text	String	The text to display as step or step title.
ExpectedMin	Variant	Supported with USS version 2.15. Displays a green led symbol if the parameters engineering value is >= the ExpectedMin and <= the ExpectedMax value. The ExpectedMax value is optional. If ExpectedMax is not specified the led will only be displayed green if the parameter engineering value equals the ExpectedMin. NOTE Satmon seems to allow usage of either the raw or the engineering value as ExpectedMin and ExpectedMax for state codes. The led display will be wrong if the raw value has been used. This might be improved if configuration information is available for all parameters.
ExpectedMax	Variant	See above.

7.4.2 USS Satmon importer version 2.12 known issues

This section lists the known deviations of USS Satmon displays compared to the original Satmon executor.

7.4.2.1 Alphanumeric Displays

- No telemetry is displayed for OutputAlphas using a SID in the Param attribute. On a few newer displays the SID is used to identify an item instead of the expected opsname. These items won't be acquired by USS and therefore not be displayed.

- Telemetry fields draw outside of their bounds making values unreadable. USS does not perform clipping on field contents like Satmon, but draws outside of the field bounds if the field is too small. Space for 2 characters is lost by default, because dqi characters are turned on for all fields. This could be improved by preparing a mcs_dqistyle for Satmon which does not have dqi characters.

- Some static text elements are colored white instead of their original color. New styles may have been used in Satmon created after June 2006 which are unknown to the importer, and therefore are colored with the default white (see Table 7.5).

- Freeze mode does not work for all Satmon displays. Workaround: Add correct SID identifier to the USS DataSource description.

7.4.2.2 Plots

- No graph is displayed if it contains discrete measurements. This could be fixed by updating all datasources with the correct type information from a XML SCOE configuration.

7.4.2.3 Procedure Displays

- Telemetry field for a procedure step is missing in all procedure displays.

- Range monitoring and indication for the parameter referenced in a procedure step is missing in all procedure displays.

- Large procedures are not displayed in multiple columns. The Satmon executor seems to wrap very long procedures into multiple columns. USS always displays procedures as one column.

7.4.3 USS Satmon importer version 2.15 known issues

This section lists the known deviations of USS Satmon displays compared to the original Satmon executor.

7.4.3.1 Alphanumeric Displays

- Telemetry field contents are clipped on the right to match their field bounds if the value is too large. A "..." character is displayed in the field if the value has been clipped. Fields with 1 character width will only display the clipping character "..." if the value is too long.

- Some static text elements are colored white instead of their original color. New styles may have been used in Satmon created after August 2007 which are unknown to the importer, and therefore are colored with the default white (see Table 7.6).

7.4.3.2 Plots

- No graph is displayed for some discrete measurements. This could be fixed by updating all datasources with the correct type information from a XML SCOE configuration. Most discrete measurements are now detected if the ops name contains "_STAT".

7.4.3.3 Procedure Displays

- Range monitoring fails for state codes that have been specified as raw value instead of their engineering value.

- Large procedures are not displayed in multiple columns. The Satmon executor seems to wrap very long procedures into multiple columns. USS always displays procedures as one column.

7.5 Updating Configuration Information in Displays

USS Displays contain configuration information for telemetry and commands from the configuration they have been created for. This information might have to be updated to make the displays usable with a different configuration. To identify the need for an update, a consistency check should be performed against an xml scoe file for the target configuration. Converted Satmon displays do not contain any configuration information and always have to be updated with configuration information.

The following tool can be used to update telemetry datasources with configuration information from an xml scoe file. Optionally a report can be written containing all the changes performed during the update. It is recommended to perform a consistency check on the resulting displays.

```
uss-update-references.[sh|bat] [OPTION(S)] ccu_int_version display_dir
```

- `OPTION`: Specify `-h` for getting a complete list of available options.
- `ccu_int_version`: The ccu internal version of the configuration to perform the update with. If the xml scoe file for the specified version is not located in the default scoe folder, a scoe file path can be specified via the `-s` option.
- `display_dir`: Directory containing the uss display files with extension .uss

Chapter 8

Localizing Displays For Different Languages

8.1 Introduction

USS provides support for localizing displays into different languages. The support includes the translation of textual element attributes.

To localize displays USS reads localization files that lie in the directory next to the displays. Those localization files contain replacement mappings in the form `key=value`, where the key describes exactly the attribute within an element that needs to be translated and the value contains the translated text.

The localization files can be generated automatically by a tool called `uss-localizer-generator`. The translation itself must be done manually by a professional translator.

Localizations can exist for many different languages in parallel and are selected by the current Locale setting set in USS.

8.2 Translation work-flow

This section describes the procedure work-flow of generating display translations.

1. Generate Display with the Editor.
2. Generate Localization skeletons.
3. Manually translate localization files.
4. Open display in Executor. Display is automatically translated according to the localization files.

8.3 Generating skeletons with the Skeleton Generator

Following command reveals the options of the skeleton generator:

```
uss-localizer-generator.bat -h
usage: uss-localizer.[sh|bat] [OPTION]... input dir
Generates locale skeleton files for the displays in source-dir.
Localize Bundles are written in the display directory.
Only supports .uss files.
OPTION may be one or more of the following.
 -c,--country          set country for resource bundles
 -f,--forceOverwrite   force overwrite of existing reports
 -h,--help             print this text and exit
 -l,--language         set language for resource bundles
 -r,--recursive        recursively descend in sub-directories
```

The generated skeletons will have the same name as the displays with the language and country abbreviation added to the name as in following example. For Display

```
COL_HOME.uss
```

the British English language file will be named

```
COL_HOME_en_GB
```

The generated files will lie in the same directory as the display.

The values in the generated files will correspond to the values set in the original display files surrounded by special marker chars marking the values as untranslated.

8.4 Supported attributes

Table 8.1: Supported attributes

Element	Translatable Attributes
Display	title
Label	text, tooltip
Button	pressedText, releasedText, tooltip
CommandList	itemNames, tooltip
ComboBox	keyNames, tooltip
Compound	all translateable properties of sub-elements, tooltip
Graph	title, defaultDomainAxisLabel, defaultRangeAxisLabel, tooltip
LineGraph	like Graph and additionally labels for all multiple DomainAxis and RangeAxis
StripGraph	Translatable properties of Graph for the graph as a whole and each included subgraph

8.5 Format of Entry Keys

The keys of the localization file uniquely determine the attribute that needs to be translated.

Keys for Display global attributes (currently only display title) have following format:

```
Display.attributeName=
```

Normal Attribute Keys have following format:

```
ElementType.elementName.attributename=
```

Container elements like Compound and StripGraph have following format to address subcomponents:

```
ContainerType.containerName.SubComponentType.subComponentName=
```

The depth of nesting is not limited for the localization files.

Chapter 9

Reference

9.1 Menu references for the executor

9.1.1 File Menu

The actions on files are grouped into the file menu.

- **Open... (Ctrl-O)** Opens a display of any format, that is USS displays, FWDU displays, GWDU displays or PREP and SATMON displays. (which are both in XML format)
  - See also: Loading display from file system
- **Save A Copy...** Saves a copy of the current display in one of the following formats: USS (*.uss), FWDU (*.fwdu), GWDU (*.gwdu) or PREP and SATMON displays (which are both saved as *.xml files).
  - See also: Saving a copy of current display
- **Close (Ctrl-W)** Closes the current display.
  - See also: Closing display
- **Close Other** Closes all displays but the currently selected one (if there is more than one display open).
  - See also: Closing other displays
- **Close All** Closes all open displays.
  - See also: Closing all displays
- **Reload (F5)** Reloads the current display.
  - See also: Reloading display from file system
- **Load Layout... (Ctrl-L)** Loads a previously saved layout which is any number of displays.
  - See also: Loading window layout
- **Save Layout As... (Ctrl-S)** Saves all open displays as a layout.
  - See also: Saving window layout
- **Create HTML Report...** Creates a HTML report of the current display which can be viewed in a browser.
- **Save Snapshot (Ctrl+Shift+P)** Save a snapshot of the current display into the user's application directory.
  - See also: Creating display snapshot
- **Print... (Ctrl+P)** Opens a print dialog to print the current display.

  - See also: Printing display

- **Print Preview...** Opens a dialog to setup the page, then opens a print preview dialog.
  - See also: Print preview
- **Properties (Alt-Enter)** Opens the properties dialog of the current display.
  - See also: Showing display properties
- **1 ... 4** The most recently opened displays.
- **Exit (Alt-X)** Exits the executor.
  - See also: Exiting the executor

9.1.2 Navigate Menu

The navigate menu groups actions for display navigation.

- **Back (Alt-Left)** Navigates to the next display to the left (if there is more than one display open).
- **Forward (Alt-Right)** Navigate to the next open display to the right (if there is more than one display open).
- **Home (F12)** Opens the home display.
  - See also: Navigating to home display
- **Find Text As You Type (Ctrl-F)** Searches for entered text in the current display. If occurrences are found they are highlighted by a surrounding orange box.
  - See also: Finding text in display
- **Find Parameter... (Ctrl+Shift-F)** Searches for a parameter (which can be a pathname, SID, PUI or opsname) in displays.
  - See also: Finding display with parameter references

9.1.3 View Menu

The view menu groups actions for the displaying or not displaying of GUI elements.

- **Toolbar** Toggles between Toolbar is displayed and not displayed.
  - See also: Showing/Hiding the toolbar
- **Show Hierarchy (F11)** Toggles between Hierarchy is shown or not shown.
  - See also: Navigating display hierarchy

9.1.4 Options Menu

The setting of preferences, the connecting to or from MCS can be done in the options menu.

- **Preferences...** Opens a dialog that allows the setting of preferences changing the behaviour of the executor e.g. tooltip behaviour, tab appearance etc.
  - See also: Configuring user settings
- **MCS Connection...** Opens a dialog in which the settings for the MCS Connection can be configured. The connection can be established or disconnected.
  - See also:
    * Connecting to system to be monitored and controlled
    * Disconnecting system to be monitored and controlled

9.1.5 Window Menu

Actions that manipulate the appearance of the windows are grouped into the window menu.

- **Cascade** Displays all open displays cascaded.
- **Tile** Displays all open displays as tiles.
- **Minimize All** Minimizes all open displays.
- **Maximize All** Maximizes all open displays.
- **Reset to default size** The current display is displayed in its default size, which has been configured by the display author.
  - See also: Resetting window to default size
- **Tabbed Mode** Toggles between displays being displayed in tabbed mode or not.
  - See also: Toggling tabbed mode
- **Undock** Undocks the current display into a standalone application.
  - See also: Undocking windows
- **Dock All** Docks all undocked displays back into the executor.
  - See also: Docking all windows
- **Show Command Response** Toggles command response visibility for a display containing tele commands.
  - See also: Showing/Hiding Command Responses

9.1.6 Help Menu

Everything that gives information about the use of USS is grouped into the help menu.

- **Help Contents** Opens the index of the user manual in HTML.
- **For Display: name of display** Shows the help for the current display.
  - See also: Showing display help
- **Search...** Opens a dialog to search the user manual for a given phrase.
- **Show All Hints** Pops a dialog up which asks if all hints should be displayed again.
- **Data Quality Indicators for Display** Displays the DQI (Data Quality Indicators) of the current display in an opened browser window.
- **About...** Opens a dialog displaying information about USS.
  - See also: Getting the executor version information

9.2 Display Versions and Compatibility

USS displays are stored in XML format with the file extension `.uss` (see Section 9.3 for the XML schema). The USS XML format is versioned independently of the USS software since the software is likely to change more frequently than the display format. It is obvious that extensions added to later versions may not be compatible with earlier versions. In order to recognize possible conflicts, every display contains a version identifier. In the example below, the format version of the display is 5.

```
<?xml version="1.0"?>
<USSObject>
  <Generator>uss-2.9.0</Generator>
  <FormatVersion>5</FormatVersion>
  <Display>
    <Width>880</Width>
    <Height>560</Height>
    ...
  </Display>
</USSObject>
```

This version identifier is checked each time the display is opened by the USS software. The general rules are:

- Backward compatibility: USS software supporting display format N can also read all displays with versions less than N. With other words: displays will never become outdated or unusable.
- No forward compatibility: USS software supporting display format N cannot read displays with versions higher than N (since it does not and can not know about the new features in the higher versions). In this case, the software issues a warning message and refuses to load the display.

In consequence, these rules mean: you can update to newer USS software and can still open all your old displays, but you can't build displays with newer USS software versions and expect them to run with old software versions.

The full list of display versions and compatibility with software versions is shown in Section 9.3.

9.3 XML Display File Format Schema

```
# Relax NG schema (http://www.relaxng.org/) for USS display files in XML format.
# $Id: //uss/main/etc/uss.rnc#89 $
# Copyright (c) 2003-2008 Astrium Space Transportation
#
# This schema defines the USS display format.
#
# This file (uss.rnc) is released as Open Source under the following
# MIT-style license.
#
# Permission is hereby granted, free of charge, to any person obtaining
# a copy of this software and associated documentation files (the
# "Software"), to deal in the Software without restriction, including
# without limitation the rights to use, copy, modify, merge, publish,
# distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so, subject to
# the following conditions:
#
# The above copyright notice and this permission notice shall be
# included in all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
# EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
# MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
# NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
# LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
# WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.


# Change History:
#
# Format 6 is written since USS-2.14.x and can be read by all newer versions.
#   - added "CLIP_LEFT" and "CLIP_RIGHT" to Field.OverflowBehavior for
#     additional clipping options
```

```
#   - TeleCommandParameter.Value is now obsolete and is no longer written
#
# Format 5 is written since USS-2.9.x  and can be read by all newer versions.
#   - added "TC" to TeleCommand.Kind for tele commands
#   - added Field.OverrideDQI and Field.FillStyle for individual coloring
#   - added Symbol.CornerEnabled, Symbol.FillStyle, Symbol.CornerFillStyle,
#     and Symbol.DQIFlagEnabled for DQI support on all symbols
#   - added Symbol.ValueMapping and Symbol.StateForInvalidValues for mapping
#     parameter values to symbol values
#   - added ComboBox.SimpleTeleCommandParameter
#
# Format 4 is written since USS-2.6.x  and can be read by all newer versions.
#
# Format 1-3 was written by internal versions of USS only and are not supported ↩
    anymore.


default namespace = ""

### Start of grammar. The remaining patterns are alphabetically ordered.

start =
  element USSObject {
    element Generator { text },
    element FormatVersion { xsd:positiveInteger },
    element Display { Display }
  }

Arc =
  element StartAngle { xsd:integer },
  element ArcAngle { xsd:integer },
  Shape

AxisProperties =
  element Owner { _GraphOrRef },
  element Label { text }?

AxisRange =
  element Lower { xsd:double },
  element Upper { xsd:double }

BarGraph =
  element DomainAxisLabel { text }?,
  element DefaultRangeAxis { CategoryAxisProperties },
  Graph

Button =
  element IsToggle { xsd:boolean },
  element IsPressed { xsd:boolean },
  element IsEnabled { xsd:boolean },
  element CornerEnabled { xsd:boolean },
  element Shape { "RECTANGLE" | "OVAL" | "FLAT" },
  element PressCommand { _CommandOrRef }?,
  element ReleaseCommand { _CommandOrRef }?,
  element PressedCompound { Compound }?,
  element ReleasedCompound { Compound }?,
  element Rotation { _RotationEnum },
  element FillStyle { FillStyle },
  element CornerFillStyle { FillStyle },
  Element

CAGShape =
  element Shapes {
    attribute class { "linked-list" },
```

```
    (
      element Arc { Arc }
    | element CAGShape { CAGShape }
    | element Ellipse { Ellipse }
    | element Polygon { Polygon }
    | element Rectangle { Rectangle }
    )+
  },
  element Operation { "UNION" | "SUBTRACT" | "INTERSECT" | "EXCLUSIVE_OR" },
  Shape

CategoryAxisProperties =
  element AxisRange { AxisRange }?,
  element AutoRange { xsd:boolean },
  element AutoRangeUpper { xsd:double }?,
  element AutoRangeLower { xsd:double }?,
  element StickyZero { xsd:boolean },
  AxisProperties

CheckValve =
  Valve

CloseDisplayCommand =
  Command

Color =
  element red { xsd:nonNegativeInteger },
  element green { xsd:nonNegativeInteger },
  element blue { xsd:nonNegativeInteger },
  element alpha { xsd:nonNegativeInteger }

ComboBox =
  element VisibleRowCount { xsd:integer },
  ValidInputMap,
  element TextStyle { TextStyle },
  element SimpleTeleCommandParameter { SimpleTeleCommandParameter | _reference  ↩
      }?,
  Element

# Command.Tooltip is obsolete since uss-2.12, but it wont't be removed
# for reasons of backward compability of *.uss file format.
Command =
  element Tooltip { text }?

CommandButton =
  element IsGuarded { xsd:boolean },
  Button

CommandList =
  element ShowButtons { xsd:boolean },
  element BackgroundColor { Color },
  element SelectionColor { Color },
  element TextStyle { TextStyle },
  element ShowAsComboBox { xsd:boolean },
  element List {
    element CommandList.Item {
      element Label { text },
      element Command { _CommandOrRef }
    }+
  },
  Element

Compound =
  element Elements { _ElementChoice* },
```

```
  Element

Computation =
  element Expression { text },
  element Arguments {
    (
      element ExternalDataSource { ExternalDataSource | _reference }
    |
      element Computation { Computation | _reference }
    )*
  },
  element StatusPropagation { xsd:boolean },
  DataSource

Criteria =
  element Key {
    (attribute class { text }?, _reference)
  |
    (
      element ExternalDataSource { ExternalDataSource | _reference }
    | element Computation { Computation | _reference }
    | element DataSourcePair { DataSourcePair | _reference }
    )
  },
  element Name { text }

DataSource =
  element Names {
    attribute class { "linked-hash-map" },
    element entry {
      element string { text },
      element string { text }
    }+
  },
  element Type {
    attribute class { text }?,
    element Literals {
      attribute class { text },
      element string { text }+
    }?,
    element Name { text }
  },
  element Unit { text }?,
  element UsingRaw { xsd:boolean }

DataSourcePair =
  (
    element DomainDataSource { _DataSourceOrRef },
    element RangeDataSource { _DataSourceOrRef }?
  )
|
  (
    element DomainDataSource { _DataSourceOrRef }?,
    element RangeDataSource { _DataSourceOrRef }
  )

Display =
  element Title { text }?,
  element Width { xsd:integer },
  element Height { xsd:integer },
  element BackgroundColor { Color }?,
  element BackgroundSymbolName { text }?,
  element BackgroundSymbolLibraryName { text }?,
  element ExecuteButton { xsd:boolean },
```

```
  element TargetSystem { text }?,
  element DatabaseAlias { text }?,
  element HelpURL { text }?,
  element Description {
    element Format { "PLAIN" | "HTML" },
    element Text { text }
  }?,
  element ChangeLog {
    element ChangeLogEntries {
      element ChangeLogEntry {
        element Revision { text },
        element Author { text },
        element Date { xsd:integer }, # seconds since the epoch
        element Comment { text }
      }*
    }
  },
  element Source {
    element Context { text }?,
    element Properties {
      element property {
        attribute name { text },
        attribute value { text }
      }*
    }
  }?,
  element Elements { _ElementChoice* }

DrawStyle =
  element Pattern { "NONE" | "SOLID" | "DOTTED" | "DASHED" },
  element Color { Color }?,
  element Width { xsd:float }

Element =
  element X { xsd:integer },
  element Y { xsd:integer },
  element Width { xsd:integer },
  element Height { xsd:integer },
  element Name { text },
  element Depth { xsd:integer },
  element Tooltip { text }?,
  element Comment { text }?,
  element DataBindings {
    attribute class { "linked-hash-set" },
    element DataBinding {
      element DataSource { _DataSourceOrRef },
      element DynamicProperty {
          "X" | "Y" | "WIDTH" | "HEIGHT"
        | "VALUE" | "DRAW_COLOR" | "FILL_COLOR" | "FLUID" | "NAVIGATION_TARGET" | ↩
            "CORNER_COLOR"
      }
    }*
  }

Ellipse =
  Shape

EllipticTickMeter =
  element StartAngle { xsd:decimal },
  element SweepAngle { xsd:decimal },
  element DrawAsCircle { xsd:boolean },
  TickMeter

ExitCommand =
```

```
  Command

ExternalDataSource =
  DataSource

ExternalImage =
  element Pathname { text },
  Image

Field =
  element Rows { xsd:integer },
  element Columns { xsd:integer },
  element Format { text }?,
  element Decimals { xsd:integer },
  element Unit { text }?,
  element ShowUnit { xsd:boolean },
  element ShowIndicators { xsd:boolean },
  element AutoWrap { xsd:boolean },
  element TextStyle { TextStyle },
  element UnitTextStyle { TextStyle },
  element Rotation { _RotationEnum },
  element OverflowBehavior { "OVERWRITE" | "EXPAND" | "SHOW_HASHES" | "CLIP_LEFT" ↩
       | "CLIP_RIGHT" },
  element OverrideDQI { xsd:boolean },
  element FillStyle { FillStyle },
  Element

FileChooser =
  element StartDirectory { text }?,
  element FilterList {
    attribute class { "linked-list" },
    element FileChooser.FileFilter {
      element Extension { text },
      element Description { text }
    }*
  },
  element UseAbsolutePath { xsd:boolean },
  InputField

FillStyle =
  element Pattern { "NONE" | "SOLID" },
  element Color { Color }

Fluid =
  element Name {
    "CUSTOM_FLUID" | "AIR" | "AMMONIA" | "BRINE"
  | "CO2" | "DISTILLATE" | "EMPTY" | "FREON" | "FUEL"
  | "HELIUM" | "HYDROGEN" | "MIXED_GAS_AND_LIQUID"
  | "NITROGEN" | "NON_TOXIC_COOLANT" | "OXIDIZER"
  | "OXYGEN" | "TOXIC_COOLANT" | "URINE" | "VACUUM" | "WATER"
  },
  element Color { Color }

Graph =
  element GraphDataset { GraphDataset }?,
  element Title { text }?,
  element Orientation { "HORIZONTAL" | "VERTICAL" },
  element LegendEnabled { xsd:boolean },
  element LegendSections { xsd:positiveInteger },
  element LegendFieldColumns { xsd:positiveInteger },
  element LegendFieldDecimals { xsd:nonNegativeInteger },
  element LegendFieldTextStyle { TextStyle }?,
  element LegendLabelTextStyle { TextStyle }?,
  element LegendUnitTextStyle { TextStyle }?,
```

```
  element CriteriaDrawStyles {
    element DrawStyle { DrawStyle }+
  }?,
  element GraphBackgroundColor { Color }?,
  element PlotBackgroundColor { Color }?,
  element DomainGridlineDrawStyle { DrawStyle }?,
  element RangeGridlineDrawStyle { DrawStyle }?,
  Element

GraphDataset =
  element Graph { _GraphOrRef }?,
  element CriteriaMap {
    attribute class { "linked-hash-map" },
    element entry {
      ( element ExternalDataSource { ExternalDataSource | _reference }
      | element Computation { Computation | _reference }
      | element DataSourcePair { DataSourcePair | _reference }
      ),
      element Criteria { Criteria }
    }*
  }

Image =
  element Rotate { xsd:decimal },
  element AutoScale { xsd:boolean },
  element KeepAspectRatio { xsd:boolean },
  Element

InputField =
  element Columns { xsd:integer },
  element TextStyle { TextStyle },
  element SimpleTeleCommandParameter { SimpleTeleCommandParameter | _reference  ↩
      }?,
  Element

Label =
  element Text { text },
  element TextStyle { TextStyle },
  element AutoSize { xsd:boolean },
  element Rotation { _RotationEnum },
  element LabelFor { _ElementOrRef }?,
  element AutoText {
    element Mode { "OFF" | "CONTEXT" | "BASENAME" },
    element Context { text }?,
    element Length { xsd:integer },
    element Clipping { "OFF" | "LEFT" | "RIGHT" },
    element ClipIndicator { text }?
  },
  Element

# Limit values must be parseable as Java Number objects

Limits =
  element LowCautionLimit { attribute class { text }, text }?,
  element HighCautionLimit { attribute class { text }, text }?,
  element LowWarningLimit { attribute class { text }, text }?,
  element HighWarningLimit { attribute class { text }, text }?,
  element LowOffScaleWarningLimit { attribute class { text }, text }?,
  element HighOffScaleWarningLimit { attribute class { text }, text }?,
  element DeltaCautionLimit { attribute class { text }, text }?,
  element DeltaWarningLimit { attribute class { text }, text }?,
  element ExpectedValue { text }?

LinearTickMeter =
```

```
  TickMeter

LineGraph =
  element ExpirationPeriod { xsd:nonNegativeInteger },
  element ExpirationSamples { xsd:nonNegativeInteger },
  element ValueMarkerEnabled { xsd:boolean },
  element StepCurveEnabled { xsd:boolean },
  element LimitAreasFilled { xsd:boolean },
  element DomainAxisLimits { Limits },
  element RangeAxisLimits { Limits },
  element DefaultDomainAxis { ValueAxisProperties },
  element DefaultRangeAxis { ValueAxisProperties },
  element DomainAxesMap { _AxesMap }?,
  element RangeAxesMap { _AxesMap }?,
  Graph

Meter =
  element Minimum { xsd:double },
  element Maximum { xsd:double },
  element Orientation { "HORIZONTAL" | "VERTICAL" },
  element BorderColor { Color }?,
  element FillColorIndicatingStatus { xsd:boolean },
  element FillStyle { FillStyle },
  Element

NavigationButton =
  element DefaultNavigationTarget { text }?,
  Button

NestedTeleCommandParameter =
  element Parameters {
    attribute class { "linked-list" },
    (
      element SimpleTeleCommandParameter { _reference }
    |
      element SimpleTeleCommandParameter { SimpleTeleCommandParameter }
    |
      element NestedTeleCommandParameter { NestedTeleCommandParameter }
    )*
  },
  TeleCommandParameter

OpenDisplayCommand =
  element DisplayBasename { text },
  element OpenInNewWindow { xsd:boolean },
  element Coordinates {
    element X { xsd:integer },
    element Y { xsd:integer },
    element Width { xsd:integer },
    element Height { xsd:integer }
  }?,
  Command

Pipe =
  element Show3DEffect { xsd:boolean },
  element PipeConnectors {
    attribute class { "linked-hash-set" },
    element PipeConnector { PipeConnector | _reference }*
  },
  element Diameter { xsd:decimal },
  element Fluid { Fluid },
  Element

PipeConnector =
```

```
  PipeSegments,
  element Point { Point }

PipeSegment =
  element PipeSegment {
    element Source {
      _reference
    },
    element Destination { PipeConnector }
  }

PipeSegments =
  element PipeSegments {
    attribute class { "linked-hash-set" },
    (PipeSegment | element PipeSegment { _reference})*
  }

PlaceHolder =
  Element

Point =
  element x { xsd:integer },
  element y { xsd:integer }

Polygon =
  element Points {
    element Point { Point }+
  },
  Shape

Polyline =
  element Points {
    element Point { Point }+
  },
  element DrawStyle { DrawStyle },
  element ArrowStart { xsd:boolean },
  element ArrowEnd { xsd:boolean },
  Element

Rectangle =
  Shape

Shape =
  element FillStyle { FillStyle },
  element DrawStyle { DrawStyle },
  Element

SimpleTeleCommandParameter =
  TeleCommandParameter

StringTeleCommand =
  TeleCommand

StripGraph =
  element SubgraphMap {
    attribute class { "linked-hash-map" },
    element entry {
      element LineGraph { LineGraph },
      element int { xsd:positiveInteger }
    }*
  },
  LineGraph

StructuredTeleCommand =
```

```
  element Parameters {
    attribute class { "linked-list" },
    (
      element SimpleTeleCommandParameter { _reference }
    |
      element SimpleTeleCommandParameter { SimpleTeleCommandParameter }
    |
      element NestedTeleCommandParameter { NestedTeleCommandParameter }
    )*
  },
  TeleCommand

Symbol =
  element LibraryName { text },
  element SymbolName { text },
  element ValueMapping {
    element entry {
      element string { text },
      element string { text }
    }*
  }?,
  element StateForInvalidValues { text }?,
  element CornerEnabled { xsd:boolean },
  element FillStyle { FillStyle },
  element CornerFillStyle { FillStyle },
  element DQIFlagEnabled { xsd:boolean },
  Image

TankMeter =
  element Fluid { Fluid },
  Meter

TeleCommand =
  element Name { text }?,
  element Kind { "FLAP" | "SWOP" | "HLCL" | "PCS" | "USS" | "TC" },
  Command

TeleCommandParameter =
  element Owner { StructuredTeleCommand | _reference },
  element Name { text },
  element Constraint { attribute class { text }?, _anyElement* }?,
  element InputOutputMapping {
    attribute class { "linked-hash-map" },
    element entry {
      element string { text },
      element string { text }
    }*
  }?,
  element Value { text }?

TextStyle =
  element Fontname { text },
  element Fontsize { xsd:nonNegativeInteger },
  element IsBold { xsd:boolean },
  element IsItalic { xsd:boolean },
  element IsUnderlined { xsd:boolean },
  element Color { Color },
  element HorizontalAlignment { "LEFT" | "CENTER" | "RIGHT" },
  element VerticalAlignment { "TOP" | "CENTER" | "BOTTOM" }

Thermometer =
  TickMeter

TickMeter =
```

```
  element TickBase { xsd:double }?,
  element Color { Color },
  element TickMajorFrequency { xsd:integer },
  element TickUnit { xsd:decimal },
  element TickIndicator {
    attribute class { text },
    element BackgroundColor { Color }?,
    element SliderStyle { "TRIANGLE" | "BAR" }?,
    element NeedleColor { Color }?,
    element NeedleStyle { "LINE" | "KITE" | "BAR" }?,
    element Thickness { xsd:integer }
  },
  element LabelStyle { "NO_LABELS" | "LEFT_OR_TOP" | "RIGHT_OR_BOTTOM" | " ↩
      ALTERNATE_START_LEFT_OR_TOP" | "ALTERNATE_START_RIGHT_OR_BOTTOM" },
  element FieldStyle { "DISABLED" | "CENTER" | "LEFT" | "RIGHT" },
  element Label { text }?,
  element LabelTextStyle { TextStyle },
  element IndicatorTextStyle { TextStyle },
  Meter

ValidInputMap =
  element ValidInputMap {
    element entry {
      element string { text },
      element string { text }
    }*
  }

ValueAxisProperties =
  element AxisMode { "VALUE_BASED_LINEAR" | "VALUE_BASED_LOGARITHMIC" | " ↩
      TIME_BASED_ABSOLUTE" | "TIME_BASED_RELATIVE" | "SAMPLE_BASED_ABSOLUTE" | " ↩
      SAMPLE_BASED_RELATIVE" },
  element AxisRange { AxisRange }?,
  element AutoRange { xsd:boolean },
  element AutoMove { xsd:boolean },
  element StickyZero { xsd:boolean },
  element TickLabelsRotated { xsd:boolean },
  AxisProperties

Valve =
  element Fluid { Fluid },
  element Modifier { "GENERIC" | "AUTO_MOTOR" | "RELIEF" | "MANUAL_MOTOR" | " ↩
      MANUAL_GENERIC" },
  element Rotation { _RotationEnum },
  element ThreeWayValve { xsd:boolean },
  Element


### Shortcuts for frequent patterns

# FIXME: this is temporary pattern accepting any XML structure until a
# full definition is provided

_anyElement =
  element * {
    (attribute * { text }
     | text
     | _anyElement)*
  }

_reference =
  attribute reference { text }

_AxesMap =
```

```
  attribute class { "linked-hash-map" },
  element entry {
    (element DataSourcePair { DataSourcePair | _reference }
    | element null { empty }),
    element ValueAxisProperties { ValueAxisProperties }
  }+

_CommandOrRef =
  CloseDisplayCommand
| (attribute class { "CloseDisplayCommand" }, (_reference | CloseDisplayCommand))
| ExitCommand
| (attribute class { "ExitCommand" }, (_reference | ExitCommand))
| OpenDisplayCommand
| (attribute class { "OpenDisplayCommand" }, (_reference | OpenDisplayCommand))
| StringTeleCommand
| (attribute class { "StringTeleCommand" }, (_reference | StringTeleCommand))
| StructuredTeleCommand
| (attribute class { "StructuredTeleCommand" }, (_reference |  ↩
    StructuredTeleCommand))

_DataSourceOrRef =
  ExternalDataSource
| (attribute class { "ExternalDataSource" }, (_reference | ExternalDataSource))
| Computation
| (attribute class { "Computation" }, (_reference | Computation))

_ElementChoice =
    element Arc { Arc | _reference }
  | element BarGraph { BarGraph | _reference }
  | element CAGShape { CAGShape | _reference }
  | element CheckValve { CheckValve | _reference }
  | element ComboBox { ComboBox | _reference }
  | element CommandButton { CommandButton | _reference }
  | element CommandList { CommandList | _reference }
  | element Compound { Compound | _reference }
  | element Ellipse { Ellipse | _reference }
  | element EllipticTickMeter { EllipticTickMeter | _reference }
  | element ExternalImage { ExternalImage | _reference }
  | element Field { Field | _reference }
  | element FileChooser { FileChooser | _reference }
  | element InputField { InputField | _reference }
  | element Label { Label | _reference }
  | element LineGraph { LineGraph | _reference }
  | element LinearTickMeter { LinearTickMeter | _reference }
  | element NavigationButton { NavigationButton | _reference }
  | element Pipe { Pipe | _reference }
  | element PlaceHolder { PlaceHolder | _reference }
  | element Polygon { Polygon | _reference }
  | element Polyline { Polyline | _reference }
  | element Rectangle { Rectangle | _reference }
  | element StripGraph { StripGraph | _reference }
  | element Symbol { Symbol | _reference }
  | element TankMeter { TankMeter | _reference }
  | element Thermometer { Thermometer | _reference }
  | element Valve { Valve | _reference }

_ElementOrRef =
  attribute class { text },
  ( _reference
  |
    ( Arc
    | BarGraph
    | CAGShape
    | CheckValve
```

```
    | ComboBox
    | CommandButton
    | CommandList
    | Compound
    | Ellipse
    | EllipticTickMeter
    | ExternalImage
    | Field
    | FileChooser
    | InputField
    | Label
    | LineGraph
    | LinearTickMeter
    | NavigationButton
    | Pipe
    | PlaceHolder
    | Polygon
    | Polyline
    | Rectangle
    | StripGraph
    | Symbol
    | TankMeter
    | Thermometer
    | Valve
    )
  )

_GraphOrRef =
  attribute class { text },
  (_reference
  |
    ( BarGraph
    | LineGraph
    | StripGraph
    )
  )

_RotationEnum =
  "DEG0" | "DEG90" | "DEG180" | "DEG270"
```

9.4 RGB Colors

List of predefined colors that can be used in computations. See Example 4.3 for what they can be used for.

```
! $Xorg: rgb.txt,v 1.3 2000/08/17 19:54:00 cpqbld Exp $
255 250 250   snow
248 248 255   ghost white
248 248 255   GhostWhite
245 245 245   white smoke
245 245 245   WhiteSmoke
220 220 220   gainsboro
255 250 240   floral white
255 250 240   FloralWhite
253 245 230   old lace
253 245 230   OldLace
250 240 230   linen
250 235 215   antique white
250 235 215   AntiqueWhite
255 239 213   papaya whip
255 239 213   PapayaWhip
255 235 205   blanched almond
```

```
255 235 205   BlanchedAlmond
255 228 196   bisque
255 218 185   peach puff
255 218 185   PeachPuff
255 222 173   navajo white
255 222 173   NavajoWhite
255 228 181   moccasin
255 248 220   cornsilk
255 255 240   ivory
255 250 205   lemon chiffon
255 250 205   LemonChiffon
255 245 238   seashell
240 255 240   honeydew
245 255 250   mint cream
245 255 250   MintCream
240 255 255   azure
240 248 255   alice blue
240 248 255   AliceBlue
230 230 250   lavender
255 240 245   lavender blush
255 240 245   LavenderBlush
255 228 225   misty rose
255 228 225   MistyRose
255 255 255   white
  0   0   0   black
 47  79  79   dark slate gray
 47  79  79   DarkSlateGray
 47  79  79   dark slate grey
 47  79  79   DarkSlateGrey
105 105 105   dim gray
105 105 105   DimGray
105 105 105   dim grey
105 105 105   DimGrey
112 128 144   slate gray
112 128 144   SlateGray
112 128 144   slate grey
112 128 144   SlateGrey
119 136 153   light slate gray
119 136 153   LightSlateGray
119 136 153   light slate grey
119 136 153   LightSlateGrey
190 190 190   gray
190 190 190   grey
211 211 211   light grey
211 211 211   LightGrey
211 211 211   light gray
211 211 211   LightGray
 25  25 112   midnight blue
 25  25 112   MidnightBlue
  0   0 128   navy
  0   0 128   navy blue
  0   0 128   NavyBlue
100 149 237   cornflower blue
100 149 237   CornflowerBlue
 72  61 139   dark slate blue
 72  61 139   DarkSlateBlue
106  90 205   slate blue
106  90 205   SlateBlue
123 104 238   medium slate blue
123 104 238   MediumSlateBlue
132 112 255   light slate blue
132 112 255   LightSlateBlue
  0   0 205   medium blue
  0   0 205   MediumBlue
```

```
 65 105 225   royal blue
 65 105 225   RoyalBlue
  0   0 255   blue
 30 144 255   dodger blue
 30 144 255   DodgerBlue
  0 191 255   deep sky blue
  0 191 255   DeepSkyBlue
135 206 235   sky blue
135 206 235   SkyBlue
135 206 250   light sky blue
135 206 250   LightSkyBlue
 70 130 180   steel blue
 70 130 180   SteelBlue
176 196 222   light steel blue
176 196 222   LightSteelBlue
173 216 230   light blue
173 216 230   LightBlue
176 224 230   powder blue
176 224 230   PowderBlue
175 238 238   pale turquoise
175 238 238   PaleTurquoise
  0 206 209   dark turquoise
  0 206 209   DarkTurquoise
 72 209 204   medium turquoise
 72 209 204   MediumTurquoise
 64 224 208   turquoise
  0 255 255   cyan
224 255 255   light cyan
224 255 255   LightCyan
 95 158 160   cadet blue
 95 158 160   CadetBlue
102 205 170   medium aquamarine
102 205 170   MediumAquamarine
127 255 212   aquamarine
  0 100   0   dark green
  0 100   0   DarkGreen
 85 107  47   dark olive green
 85 107  47   DarkOliveGreen
143 188 143   dark sea green
143 188 143   DarkSeaGreen
 46 139  87   sea green
 46 139  87   SeaGreen
 60 179 113   medium sea green
 60 179 113   MediumSeaGreen
 32 178 170   light sea green
 32 178 170   LightSeaGreen
152 251 152   pale green
152 251 152   PaleGreen
  0 255 127   spring green
  0 255 127   SpringGreen
124 252   0   lawn green
124 252   0   LawnGreen
  0 255   0   green
127 255   0   chartreuse
  0 250 154   medium spring green
  0 250 154   MediumSpringGreen
173 255  47   green yellow
173 255  47   GreenYellow
 50 205  50   lime green
 50 205  50   LimeGreen
154 205  50   yellow green
154 205  50   YellowGreen
 34 139  34   forest green
 34 139  34   ForestGreen
```

```
107 142  35   olive drab
107 142  35   OliveDrab
189 183 107   dark khaki
189 183 107   DarkKhaki
240 230 140   khaki
238 232 170   pale goldenrod
238 232 170   PaleGoldenrod
250 250 210   light goldenrod yellow
250 250 210   LightGoldenrodYellow
255 255 224   light yellow
255 255 224   LightYellow
255 255   0   yellow
255 215   0     gold
238 221 130   light goldenrod
238 221 130   LightGoldenrod
218 165  32   goldenrod
184 134  11   dark goldenrod
184 134  11   DarkGoldenrod
188 143 143   rosy brown
188 143 143   RosyBrown
205  92  92   indian red
205  92  92   IndianRed
139  69  19   saddle brown
139  69  19   SaddleBrown
160  82  45   sienna
205 133  63   peru
222 184 135   burlywood
245 245 220   beige
245 222 179   wheat
244 164  96   sandy brown
244 164  96   SandyBrown
210 180 140   tan
210 105  30   chocolate
178  34  34   firebrick
165  42  42   brown
233 150 122   dark salmon
233 150 122   DarkSalmon
250 128 114   salmon
255 160 122   light salmon
255 160 122   LightSalmon
255 165   0   orange
255 140   0   dark orange
255 140   0   DarkOrange
255 127  80   coral
240 128 128   light coral
240 128 128   LightCoral
255  99  71   tomato
255  69   0   orange red
255  69   0   OrangeRed
255   0   0   red
255 105 180   hot pink
255 105 180   HotPink
255  20 147   deep pink
255  20 147   DeepPink
255 192 203   pink
255 182 193   light pink
255 182 193   LightPink
219 112 147   pale violet red
219 112 147   PaleVioletRed
176  48  96   maroon
199  21 133   medium violet red
199  21 133   MediumVioletRed
208  32 144   violet red
208  32 144   VioletRed
```

```
255   0 255   magenta
238 130 238   violet
221 160 221   plum
218 112 214   orchid
186  85 211   medium orchid
186  85 211   MediumOrchid
153  50 204   dark orchid
153  50 204   DarkOrchid
148   0 211   dark violet
148   0 211   DarkViolet
138  43 226   blue violet
138  43 226   BlueViolet
160  32 240   purple
147 112 219   medium purple
147 112 219   MediumPurple
216 191 216   thistle
255 250 250   snow1
238 233 233   snow2
205 201 201   snow3
139 137 137   snow4
255 245 238   seashell1
238 229 222   seashell2
205 197 191   seashell3
139 134 130   seashell4
255 239 219   AntiqueWhite1
238 223 204   AntiqueWhite2
205 192 176   AntiqueWhite3
139 131 120   AntiqueWhite4
255 228 196   bisque1
238 213 183   bisque2
205 183 158   bisque3
139 125 107   bisque4
255 218 185   PeachPuff1
238 203 173   PeachPuff2
205 175 149   PeachPuff3
139 119 101   PeachPuff4
255 222 173   NavajoWhite1
238 207 161   NavajoWhite2
205 179 139   NavajoWhite3
139 121  94   NavajoWhite4
255 250 205   LemonChiffon1
238 233 191   LemonChiffon2
205 201 165   LemonChiffon3
139 137 112   LemonChiffon4
255 248 220   cornsilk1
238 232 205   cornsilk2
205 200 177   cornsilk3
139 136 120   cornsilk4
255 255 240   ivory1
238 238 224   ivory2
205 205 193   ivory3
139 139 131   ivory4
240 255 240   honeydew1
224 238 224   honeydew2
193 205 193   honeydew3
131 139 131   honeydew4
255 240 245   LavenderBlush1
238 224 229   LavenderBlush2
205 193 197   LavenderBlush3
139 131 134   LavenderBlush4
255 228 225   MistyRose1
238 213 210   MistyRose2
205 183 181   MistyRose3
139 125 123   MistyRose4
```

```
240 255 255   azure1
224 238 238   azure2
193 205 205   azure3
131 139 139   azure4
131 111 255   SlateBlue1
122 103 238   SlateBlue2
105  89 205   SlateBlue3
 71  60 139   SlateBlue4
 72 118 255   RoyalBlue1
 67 110 238   RoyalBlue2
 58  95 205   RoyalBlue3
 39  64 139   RoyalBlue4
  0   0 255   blue1
  0   0 238   blue2
  0   0 205   blue3
  0   0 139   blue4
 30 144 255   DodgerBlue1
 28 134 238   DodgerBlue2
 24 116 205   DodgerBlue3
 16  78 139   DodgerBlue4
 99 184 255   SteelBlue1
 92 172 238   SteelBlue2
 79 148 205   SteelBlue3
 54 100 139   SteelBlue4
  0 191 255   DeepSkyBlue1
  0 178 238   DeepSkyBlue2
  0 154 205   DeepSkyBlue3
  0 104 139   DeepSkyBlue4
135 206 255   SkyBlue1
126 192 238   SkyBlue2
108 166 205   SkyBlue3
 74 112 139   SkyBlue4
176 226 255   LightSkyBlue1
164 211 238   LightSkyBlue2
141 182 205   LightSkyBlue3
 96 123 139   LightSkyBlue4
198 226 255   SlateGray1
185 211 238   SlateGray2
159 182 205   SlateGray3
108 123 139   SlateGray4
202 225 255   LightSteelBlue1
188 210 238   LightSteelBlue2
162 181 205   LightSteelBlue3
110 123 139   LightSteelBlue4
191 239 255   LightBlue1
178 223 238   LightBlue2
154 192 205   LightBlue3
104 131 139   LightBlue4
224 255 255   LightCyan1
209 238 238   LightCyan2
180 205 205   LightCyan3
122 139 139   LightCyan4
187 255 255   PaleTurquoise1
174 238 238   PaleTurquoise2
150 205 205   PaleTurquoise3
102 139 139   PaleTurquoise4
152 245 255   CadetBlue1
142 229 238   CadetBlue2
122 197 205   CadetBlue3
 83 134 139   CadetBlue4
  0 245 255   turquoise1
  0 229 238   turquoise2
  0 197 205   turquoise3
  0 134 139   turquoise4
```

```
  0 255 255   cyan1
  0 238 238   cyan2
  0 205 205   cyan3
  0 139 139   cyan4
151 255 255   DarkSlateGray1
141 238 238   DarkSlateGray2
121 205 205   DarkSlateGray3
 82 139 139   DarkSlateGray4
127 255 212   aquamarine1
118 238 198   aquamarine2
102 205 170   aquamarine3
 69 139 116   aquamarine4
193 255 193   DarkSeaGreen1
180 238 180   DarkSeaGreen2
155 205 155   DarkSeaGreen3
105 139 105   DarkSeaGreen4
 84 255 159   SeaGreen1
 78 238 148   SeaGreen2
 67 205 128   SeaGreen3
 46 139  87   SeaGreen4
154 255 154   PaleGreen1
144 238 144   PaleGreen2
124 205 124   PaleGreen3
 84 139  84   PaleGreen4
  0 255 127   SpringGreen1
  0 238 118   SpringGreen2
  0 205 102   SpringGreen3
  0 139  69   SpringGreen4
  0 255   0   green1
  0 238   0   green2
  0 205   0   green3
  0 139   0   green4
127 255   0   chartreuse1
118 238   0   chartreuse2
102 205   0   chartreuse3
 69 139   0   chartreuse4
192 255  62   OliveDrab1
179 238  58   OliveDrab2
154 205  50   OliveDrab3
105 139  34   OliveDrab4
202 255 112   DarkOliveGreen1
188 238 104   DarkOliveGreen2
162 205  90   DarkOliveGreen3
110 139  61   DarkOliveGreen4
255 246 143   khaki1
238 230 133   khaki2
205 198 115   khaki3
139 134  78   khaki4
255 236 139   LightGoldenrod1
238 220 130   LightGoldenrod2
205 190 112   LightGoldenrod3
139 129  76   LightGoldenrod4
255 255 224   LightYellow1
238 238 209   LightYellow2
205 205 180   LightYellow3
139 139 122   LightYellow4
255 255   0   yellow1
238 238   0   yellow2
205 205   0   yellow3
139 139   0   yellow4
255 215   0   gold1
238 201   0   gold2
205 173   0   gold3
139 117   0   gold4
```

```
255 193  37   goldenrod1
238 180  34   goldenrod2
205 155  29   goldenrod3
139 105  20   goldenrod4
255 185  15   DarkGoldenrod1
238 173  14   DarkGoldenrod2
205 149  12   DarkGoldenrod3
139 101   8   DarkGoldenrod4
255 193 193   RosyBrown1
238 180 180   RosyBrown2
205 155 155   RosyBrown3
139 105 105   RosyBrown4
255 106 106   IndianRed1
238  99  99   IndianRed2
205  85  85   IndianRed3
139  58  58   IndianRed4
255 130  71   sienna1
238 121  66   sienna2
205 104  57   sienna3
139  71  38   sienna4
255 211 155   burlywood1
238 197 145   burlywood2
205 170 125   burlywood3
139 115  85   burlywood4
255 231 186   wheat1
238 216 174   wheat2
205 186 150   wheat3
139 126 102   wheat4
255 165  79   tan1
238 154  73   tan2
205 133  63   tan3
139  90  43   tan4
255 127  36   chocolate1
238 118  33   chocolate2
205 102  29   chocolate3
139  69  19   chocolate4
255  48  48   firebrick1
238  44  44   firebrick2
205  38  38   firebrick3
139  26  26   firebrick4
255  64  64   brown1
238  59  59   brown2
205  51  51   brown3
139  35  35   brown4
255 140 105   salmon1
238 130  98   salmon2
205 112  84   salmon3
139  76  57   salmon4
255 160 122   LightSalmon1
238 149 114   LightSalmon2
205 129  98   LightSalmon3
139  87  66   LightSalmon4
255 165   0   orange1
238 154   0   orange2
205 133   0   orange3
139  90   0   orange4
255 127   0   DarkOrange1
238 118   0   DarkOrange2
205 102   0   DarkOrange3
139  69   0   DarkOrange4
255 114  86   coral1
238 106  80   coral2
205  91  69   coral3
139  62  47   coral4
```

```
255  99  71   tomato1
238  92  66   tomato2
205  79  57   tomato3
139  54  38   tomato4
255  69   0   OrangeRed1
238  64   0   OrangeRed2
205  55   0   OrangeRed3
139  37   0   OrangeRed4
255   0   0   red1
238   0   0   red2
205   0   0   red3
139   0   0   red4
255  20 147   DeepPink1
238  18 137   DeepPink2
205  16 118   DeepPink3
139  10  80   DeepPink4
255 110 180   HotPink1
238 106 167   HotPink2
205  96 144   HotPink3
139  58  98   HotPink4
255 181 197   pink1
238 169 184   pink2
205 145 158   pink3
139  99 108   pink4
255 174 185   LightPink1
238 162 173   LightPink2
205 140 149   LightPink3
139  95 101   LightPink4
255 130 171   PaleVioletRed1
238 121 159   PaleVioletRed2
205 104 137   PaleVioletRed3
139  71  93   PaleVioletRed4
255  52 179   maroon1
238  48 167   maroon2
205  41 144   maroon3
139  28  98   maroon4
255  62 150   VioletRed1
238  58 140   VioletRed2
205  50 120   VioletRed3
139  34  82   VioletRed4
255   0 255   magenta1
238   0 238   magenta2
205   0 205   magenta3
139   0 139   magenta4
255 131 250   orchid1
238 122 233   orchid2
205 105 201   orchid3
139  71 137   orchid4
255 187 255   plum1
238 174 238   plum2
205 150 205   plum3
139 102 139   plum4
224 102 255   MediumOrchid1
209  95 238   MediumOrchid2
180  82 205   MediumOrchid3
122  55 139   MediumOrchid4
191  62 255   DarkOrchid1
178  58 238   DarkOrchid2
154  50 205   DarkOrchid3
104  34 139   DarkOrchid4
155  48 255   purple1
145  44 238   purple2
125  38 205   purple3
 85  26 139   purple4
```

```
171 130 255   MediumPurple1
159 121 238   MediumPurple2
137 104 205   MediumPurple3
 93  71 139   MediumPurple4
255 225 255   thistle1
238 210 238   thistle2
205 181 205   thistle3
139 123 139   thistle4
  0   0   0   gray0
  0   0   0   grey0
  3   3   3   gray1
  3   3   3   grey1
  5   5   5   gray2
  5   5   5   grey2
  8   8   8   gray3
  8   8   8   grey3
 10  10  10     gray4
 10  10  10     grey4
 13  13  13     gray5
 13  13  13     grey5
 15  15  15     gray6
 15  15  15     grey6
 18  18  18     gray7
 18  18  18     grey7
 20  20  20     gray8
 20  20  20     grey8
 23  23  23     gray9
 23  23  23     grey9
 26  26  26     gray10
 26  26  26     grey10
 28  28  28     gray11
 28  28  28     grey11
 31  31  31     gray12
 31  31  31     grey12
 33  33  33     gray13
 33  33  33     grey13
 36  36  36     gray14
 36  36  36     grey14
 38  38  38     gray15
 38  38  38     grey15
 41  41  41     gray16
 41  41  41     grey16
 43  43  43     gray17
 43  43  43     grey17
 46  46  46     gray18
 46  46  46     grey18
 48  48  48     gray19
 48  48  48     grey19
 51  51  51     gray20
 51  51  51     grey20
 54  54  54     gray21
 54  54  54     grey21
 56  56  56     gray22
 56  56  56     grey22
 59  59  59     gray23
 59  59  59     grey23
 61  61  61     gray24
 61  61  61     grey24
 64  64  64     gray25
 64  64  64     grey25
 66  66  66     gray26
 66  66  66     grey26
 69  69  69     gray27
 69  69  69     grey27
```

```
 71  71  71     gray28
 71  71  71     grey28
 74  74  74     gray29
 74  74  74     grey29
 77  77  77     gray30
 77  77  77     grey30
 79  79  79     gray31
 79  79  79     grey31
 82  82  82     gray32
 82  82  82     grey32
 84  84  84     gray33
 84  84  84     grey33
 87  87  87     gray34
 87  87  87     grey34
 89  89  89     gray35
 89  89  89     grey35
 92  92  92     gray36
 92  92  92     grey36
 94  94  94     gray37
 94  94  94     grey37
 97  97  97     gray38
 97  97  97     grey38
 99  99  99     gray39
 99  99  99     grey39
102 102 102     gray40
102 102 102     grey40
105 105 105     gray41
105 105 105     grey41
107 107 107     gray42
107 107 107     grey42
110 110 110     gray43
110 110 110     grey43
112 112 112     gray44
112 112 112     grey44
115 115 115     gray45
115 115 115     grey45
117 117 117     gray46
117 117 117     grey46
120 120 120     gray47
120 120 120     grey47
122 122 122     gray48
122 122 122     grey48
125 125 125     gray49
125 125 125     grey49
127 127 127     gray50
127 127 127     grey50
130 130 130     gray51
130 130 130     grey51
133 133 133     gray52
133 133 133     grey52
135 135 135     gray53
135 135 135     grey53
138 138 138     gray54
138 138 138     grey54
140 140 140     gray55
140 140 140     grey55
143 143 143     gray56
143 143 143     grey56
145 145 145     gray57
145 145 145     grey57
148 148 148     gray58
148 148 148     grey58
150 150 150     gray59
150 150 150     grey59
```

```
153 153 153     gray60
153 153 153     grey60
156 156 156     gray61
156 156 156     grey61
158 158 158     gray62
158 158 158     grey62
161 161 161     gray63
161 161 161     grey63
163 163 163     gray64
163 163 163     grey64
166 166 166     gray65
166 166 166     grey65
168 168 168     gray66
168 168 168     grey66
171 171 171     gray67
171 171 171     grey67
173 173 173     gray68
173 173 173     grey68
176 176 176     gray69
176 176 176     grey69
179 179 179     gray70
179 179 179     grey70
181 181 181     gray71
181 181 181     grey71
184 184 184     gray72
184 184 184     grey72
186 186 186     gray73
186 186 186     grey73
189 189 189     gray74
189 189 189     grey74
191 191 191     gray75
191 191 191     grey75
194 194 194     gray76
194 194 194     grey76
196 196 196     gray77
196 196 196     grey77
199 199 199     gray78
199 199 199     grey78
201 201 201     gray79
201 201 201     grey79
204 204 204     gray80
204 204 204     grey80
207 207 207     gray81
207 207 207     grey81
209 209 209     gray82
209 209 209     grey82
212 212 212     gray83
212 212 212     grey83
214 214 214     gray84
214 214 214     grey84
217 217 217     gray85
217 217 217     grey85
219 219 219     gray86
219 219 219     grey86
222 222 222     gray87
222 222 222     grey87
224 224 224     gray88
224 224 224     grey88
227 227 227     gray89
227 227 227     grey89
229 229 229     gray90
229 229 229     grey90
232 232 232     gray91
232 232 232     grey91
```

```
235 235 235     gray92
235 235 235     grey92
237 237 237     gray93
237 237 237     grey93
240 240 240     gray94
240 240 240     grey94
242 242 242     gray95
242 242 242     grey95
245 245 245     gray96
245 245 245     grey96
247 247 247     gray97
247 247 247     grey97
250 250 250     gray98
250 250 250     grey98
252 252 252     gray99
252 252 252     grey99
255 255 255     gray100
255 255 255     grey100
169 169 169   dark grey
169 169 169   DarkGrey
169 169 169   dark gray
169 169 169   DarkGray
0     0 139   dark blue
0     0 139   DarkBlue
0   139 139   dark cyan
0   139 139   DarkCyan
139   0 139   dark magenta
139   0 139   DarkMagenta
139   0   0   dark red
139   0   0   DarkRed
144 238 144   light green
144 238 144   LightGreen
```

Appendix A

Example Configuration in MCS Environment

This appendix describes an example setting for the MCS Facility. The current setting may change without notice.

A.1 Editor Parameters

For using the USS Editor on MCS use a configuration like the one described below:

1. Choose **Edit** > **Preferences** > **Project** and set the following:

   Root folder `/san1/mcs/shared/home/cgsadmin/uss-project`

2. Choose **Database** and set the following:

   Database User ops$cgsadmin

   Database sid oracle

   Database port 1521

   Database password admin_1

   Database hostname mcs-dbs

3. Choose **MDB-General** and set the following:

   Use CDU preferred option is off (if selected MDB-CDU Preferences will be used, else MDB-CCU Preferences will be used)

   System (Tree) version 12

   System mission MASTER

   System element configuration APM

4. Choose **MDB-CDU** and set the following:

   CU version 1

   CU test Version 13

   CU revision 0

   CU path \APM\COL_CC\MCS_DEV\COL_GND_SYNOPT

   CU issue 1

   CU instance MCS_AIV1

   CU domain CGS

NOTE

MDB identification sequence is: version / issue / revision

5. Choose **MDB-CCU** and set the following:

   CU version 7

   CU revision 1

   CU path \APM

   CU name MCS_AIV

   CU issue 1

NOTE

MDB identification sequence is: version / issue / revision

6. Choose **MDB-SCOE** and set the following:

   SID 0

   File path `/san1/mcs/shared/mcs_home/uss/share/scoe/316989981_0_info.xml`

   CU internal version 316989981

NOTE

CCU/CDU preference setting should not be in conflict with SCOE file preference selection!

A.2 Executor Parameters

Set configuration like the following to use the USS Executor on MCS equipment:

1. Choose **Option** > **Preferences** > **Project** and set the following:

   Projects folder `/san1/mcs/shared/home/cgsadmin/uss-project`

   Home Display should be selected within the Project folder. The Projects folder setting is needed first to set the home display

2. Choose **Option** > **MCS Connection** and set the following:

   Service `CIS`

   Host `localhost` or `mcs-ctm` (configuration dependent)

   Port `7060`

NOTE

Within MCS, the CIS connection parameter should not contradict the configuration settings for MCS tools.

Appendix B

How to Interface with new Target Systems

This appendix describes how a programmer can implement an adapter to a new TM/TC target system.

B.1 Introduction

The USS executor is delivered by default with an interface to CGS-based systems. However, USS also provides a general extension mechanism that allows adaptation to arbitrary target systems. For this, a custom system adapter needs to be implemented. This adapter may connect to local or remote systems for receiving telemetry and sending commands.

For the implementation of a custom adapter a Java development environment is needed. An example project is provided as starting point for development. The next sections will guide you through the steps of setting up a development environment and compiling the example project. The last section explains the provided example based on snippets from the source code.

B.2 Development Environment

For the development of a custom system adapter a Java 6 based development environment is needed. This section describes the components of the development environment and their installation. Note that the following examples show output from a Linux system. However, the basic procedures are the same on Windows.

B.2.1 Prerequisites

To start development of an adapter for a new target system the following software is required:

1. A USS distribution archive version 2.19.0 or higher.
2. A USS examples archive version 2.19.0 or higher.
3. A Java 6 based Java Development Kit (JDK).
4. Apache Ant build tool version 1.6.5 or higher.

B.2.2 Setup of the Development Environment

To install the development environment perform the following steps:

1. Extract the USS distribution archive to a location suited for development (see Chapter 1 for details). It is recommended to use a location that is usually not be used for a regular installation.
2. Extract the USS examples archive into the same directory used in the previous step.
3. Install a Java Development Kit 1.6 available from Sun Microsystems

4. Install Apache Ant build tool available from The Apache Ant Project Site. This usually involves setting the `JAVA_HOME` and `ANT_HOME` environment variables to point to their respective installation directories (see Apache Ant manual for details).

You can now verify that the development environment is setup correctly by performing a first build of the provided example project. To build the example project do the following.

1. Open a shell and navigate to the example project root directory. If the USS installation directory is `$basedir`, then the example projects location is `$basedir/uss-x.y.z/examples/tutorial/systemadapter` where x,y,z are version numbers. Change the working directory to `$basedir/uss-x.y.z/examples/tutorial/systemadapter`.

2. Verify that the folder contains the following file and directory structure:

```
systemadapter
+---etc
|
+---src
|
+---build.xml
|
+---README
```

B.2.3 Compiling the Example Adapter

To compile the example adapter execute `ant` on the command line. The console output should look like the following (paths will be different depending on where you unpacked the distribution):

```
$ ant
Buildfile: build.xml

compile:
    [mkdir] Created dir: /home/mr/tmp/uss-2.19.0pre/examples/tutorial/ ↩
       systemadapter/build/classes
    [javac] Compiling 3 source files to /home/mr/tmp/uss-2.19.0pre/examples/ ↩
       tutorial/systemadapter/build/classes
    [javac] Note: /home/mr/tmp/uss-2.19.0pre/examples/tutorial/systemadapter/src/ ↩
       adapter/ExampleSystemAdapterImpl.java uses unchecked or unsafe operations ↩
       .
    [javac] Note: Recompile with -Xlint:unchecked for details.

jar:
      [jar] Building jar: /home/mr/tmp/uss-2.19.0pre/examples/tutorial/ ↩
         systemadapter/build/examplesystemadapter.jar

build:

BUILD SUCCESSFUL
Total time: 1 second
```

B.2.4 Installing and Using the Example Adapter

This section explains how to install and how to work with the example adapter.

B.2.4.1 Installing the Example Adapter

The example adapter has to be installed and activated to make it usable with the USS executor. To install the adapter the following steps are required:

1. Type `antinstall` in the shell.

```
$ ant install
```

The compiled adapter library is copied into the `$basedir/uss-x.y.z/lib/ext` directory.

> **NOTE**
>
>
>
> This step has to be performed again after modifying the example source code to re-install the adapter with your modifications. otherwise the old version will stay in effect when running the USS executor.

2. Open the file `$basedir/uss-x.y.z/etc/uss.properties` in an editor. Change the property "uss.external.systemadapter" to "adapter.ExampleSystemAdapterImpl". This step is only required once.

B.2.4.2 Running USS with the Example Adapter

After installation of the adapter, start the `uss-executor` to verify if the adapter is installed correctly.

```
$ ../../../bin/uss-executor.sh
```

Open a display in the executor that contains telemetry items eg. the `Peformance1.uss"` display located in the `$basedir/uss-x.y.z/examples/uss/demo` directory. Immediately after opening the display, values should be visible in the displays data fields. These values in the range of -10 to 10 are generated by the example dataprovider.

Now open a display in the executor that contains command buttons eg. the `Commanding.uss"` display located in the `$basedir/uss-x.y.z/examples/uss/demo` directory. On the commanding display click the top right button "Issue cmd" to trigger exection of a command via the example command handler. An information dialog should open that displays a message which starts with "ExampleCommandHandler.issueStringTelecommand". This dialog is created by the example command handler.

If you have successfully performed the previous steps, the example system adapter is installed and works correctly.

B.2.5 The Example Explained

This section explains the provided example based on snippets from the source code.

B.2.5.1 Introduction

The source code for the example is located in the `$basedir/uss-x.y.z/examples/tutorial/systemadapter/src` directory. The example consists of the 3 following classes which can be found in separate sub-folders:

1. `adapter/ExampleSystemAdapterImpl.java` Responsible for the handling of the connection state and the creation of the DataProvider and the CommandHandler.

2. `commandhandler/ExampleCommandHandler.java` Handles the transmission of commands and their responses. The example implements a simple command handler that opens a dialog whenever a command is triggered on a display.

3. `dataprovider/ExampleDataProvider.java` Handles the reception and conversion of telemetry for a single display. The example implements a simple data generator that creates a sin value between -10 and 10 for each telemetry item in a display.

The extension API consists of three interface classes for which the classes above are concrete example implementations. The example should be seen as an introduction to these interfaces. The following sections give an overview of all the methods found in the interface classes with an implementation example.

B.2.5.2 ExternalSystemAdapterInterface

The SystemAdapter is responsible for the handling of the connection state and the creation of the DataProvider and the CommandHandler. A single instance of the adapter is created on startup of the USS executor. The executor retrieves the adapter class name from the properties file `$basedir/uss-x.y.z/etc/uss.properties` via a property named "uss.external.systemadapter".

An ExternalSystemAdapterInterface implementation has to provide methods for the interface `net.eads.space.uss.shared.extern.ExternalSystemAdapterInterface`:

```
public interface ExternalSystemAdapterInterface {
    boolean connect(boolean askForConnectionParameter);
    boolean disConnect();
    boolean isConnected();
    DataProvider getDataProvider(Display display);
    HistoryService<Name, Time, Data> getDataHistoryService();
    TeleCommandHandler getTeleCommandHandler();
    String getName();
}
```

The following source code snippet from the `ExampleSystemAdapterImpl` shows an implementation example for these methods:

```
public class ExampleSystemAdapterImpl implements ExternalSystemAdapterInterface {

    /**
     * Does all the necessary steps to connect the USS-client to the external ↩
         System.
     * This might include User Interactions.
     */
    public boolean connect(boolean askForConnectionParameter) {
        // This adapter is always connected.
        // You can use a dialog here asking for connection parameters and ↩
            initialize the connection.
        // If this method returns true then calls to isConnected() should also ↩
            return true.
        return true;
    }

    /**
     * disconnects the USS-client from the external System.
     */
    public boolean disConnect() {
        // This adapter can not be disconnected so return false
        return false;
    }

    /**
     * Returns true, if the USS-client is connected to the external System.
     */
    public boolean isConnected() {
        // Always return true
        return true;
    }

    /**
     * Returns the system data provider.
     */
    public DataProvider getDataProvider(Display display) {
        // Note that A new DataProvider has to be created for each display.
        return new ExampleDataProvider(display);
    }

    /**
     * Returns the history service for this adapter or null if no history service ↩
         is
```

```
     * available.
     */
    public HistoryService<Name, Time, Data> getDataHistoryService() {
        // A service providing historic data is optional.
        return null;
    }

    /**
     * Returns the system command handler.
     */
    public TeleCommandHandler getTeleCommandHandler() {
        // You can re-use a single CommandHandler reference here instead of
            creating a new one on every call.
        return new ExampleCommandHandler();
    }

    /**
     * Returns a short descriptive name for the external System (e.g. MCS).
     */
    public String getName() {
        return "ExampleAdapter";
    }
}
```

The following table shows on which events these methods are called by the USS executor.

Table B.1: ExternalSystemAdapterInterface Methods and their Call Events

Method	Event
connect	called when connect is triggered on the uss-executor user interface
disconnect	called when disconnect is triggered on the uss-executor user interface
isconnected	can be called at any time to verify the connection state
getDataProvider	called when a new display is opened
getDataHistoryService	called when a new display is opened
getTeleCommandHandler	called when a new display is opened
getName	currently not used

B.2.5.3 DataProvider

A DataProvider is responsible for handling the subscription of telemetry items as well as their reception and translation. A new instance is created for every display that is opened in the USS executor. A DataProvider implementation has to contain all methods from the interface `net.eads.space.uss.shared.data.DataProvider`:

```
public interface DataProvider {
    void subscribe(Set< ? extends DataSource> dataSources);
    void startUpdates();
    void stopUpdates();
    String getCalibrationRules(DataSource ds);
    HistoryService<Name, Time, Data> getDataHistoryService();
}
```

The following source code snippet from the `ExampleDataProvider` shows an implementation example for these methods:

```
public class ExampleDataProvider implements DataProvider, Runnable {
    ...
```

```
    /**
     * Subscribes a set of data sources for which the callee should provide
     * data during execution.
     *
     * @param dataSources the set of data sources to retrieve data from.
     */
    public void subscribe(Set< ? extends DataSource> dataSources) {
        for (Object obj : dataSources) {
            // only animate external DataSources
            if (obj instanceof ExternalDataSource) {
                DataSource ds = (DataSource) obj;
                // multiple dataSources for the same Name are only added once
                fDataSources.put(ds.getPreferredName(), ds);
            }
        }
    }

    /**
     * Lets the data provider start sending updates. This method must ↩
         synchronously call
     * {@link net.eads.space.uss.shared.display.Display#submitData(Name, Data)} ↩
         by using
     * the caller thread to initialize all data sources with their defaults.
     */
    public synchronized void startUpdates() {
        start();
    }

    /**
     * {@inheritDoc}
     * @see net.eads.space.uss.shared.data.DataProvider#stopUpdates()
     */
    public synchronized void stopUpdates() {
        stop();
    }

    /**
     * Gets calibration rules as plain text.
     *
     * @param ds data source to get the rules for
     * @return String calibration rule or null if not available.
     */
    public String getCalibrationRules(final DataSource ds) {
        return null;
    }

    /**
     * Gets the data history service of this data provider.
     *
     * @return the data history service or null if not available.
     */
    public HistoryService<Name, Time, Data> getDataHistoryService() {
        return null;
    }
}
```

The following table shows on which events these methods are called by the USS executor.

Table B.2: DataProvider Methods and their Call Events

Method	**Event**
subscribe	called by the display with a set of datasources before a display starts execution

Table B.2: (continued)

Method	Event
startUpdates	called after call to subscribe to indicate that the display is ready to receive data
stopUpdates	called after call to startUpdates to indicate that the DataProvider should stop submitting data
getCalibrationRules	called when a properties dialog is opened for a telemetry item
getDataHistoryService	called when the DataProvider is set on a display

B.2.5.3.1 Submitting Data to the Display The methods listed previously are called by the display and largely reflect the display's lifecycle. After a display started execution and called *startUpdates* the DataProvider may submit data to the display. The following source code example shows how to create a USS Data object with status and submit it to the display:

```
...

// retrieve a DataSource from the Set that has been provided as parameter on
    the subscribe call
DataSource ds = ...
Name name = ds.getPreferredName();

// get the raw and and engineering value for the item identified by the name
double rawValue = ...;
double engValue = ...;

// creates a nominal status to attach to the data
// The status identifiers used can be defined as suitable for the target
// system. The actual status is flagged by the 6 boolean parameters.
Status status = new Status("ACQUIRED", "DISABLED", "DISABLED", "PROCESSED"
    ,true, true, true, false, false, false);

// The actual data to be send to the receiver. It consists of the status,
// the raw and engineering values, apid, packet id, Limits and
// a timestamp.
DefaultData d = new DefaultData(status, engValue, rawValue,
Integer.valueOf(0), Integer.valueOf(0),
Limits.NO_LIMITS, TimeKeeper.getYoungestSystemTime());
// submit the data to the receiving display
display.submitData(name, d);

...
```

In the previous example a single telemetry item in a display is updated by using the reference to the items DataSource. All DataSources used in a display are given to the DataProvider as Set on call of the *subscribe* method. A DataSource has a Name that identifies an item in the target system. In this example it is assumed that data for the item with this name has previously been received. An update of the display is prepared by creating a *Status* object that represents a nominal data status. In the next step a *DefaultData* object is created that contains the raw and engineering values from an update allong with the nominal status and a time stamp. The Data object is then submitted to the display.

B.2.5.4 The Telecommand Handler

The TeleCommandHandler is responsabe for handling the parsing and transmission of telecommands as well as the handling of command responses. Whenever an interactive command element is triggered the telecommand associated with the element is forwarded to the telecommand handler for execution on the target system.

A TeleCommandHandler implementation has to contain all methods from the interface `net.eads.space.uss.shared.extern.TeleCommandHandler`:

```
public interface TeleCommandHandler {
    CommandResponse issueStringTeleCommand(StringTeleCommand cmd, Display display ↩
        );
    CommandResponse issueStructuredTeleCommand(StructuredTeleCommand cmd, Display ↩
         display);
    boolean confirmCommandExecution(String title, String message,TeleCommand  ↩
        teleCommand, String targetSystem);
    String getConfirmationText(TeleCommand teleCommand, String targetSystem);
    String getEnhancedConfirmationText(TeleCommand teleCommand, String  ↩
        targetSystem,
            List<SimpleTeleCommandParameter> assocParameters);
}
```

The following source code snippet from the `ExampleCommandHandler` shows an implementation example for these methods:

```
public class ExampleCommandHandler implements TeleCommandHandler {
    ...

    /**
     * Handle execution of a StringTeleCommand. The given cmd contains the  ↩
         command string
     * that is to be executed or send to a target system.
     */
    public CommandResponse issueStringTeleCommand(StringTeleCommand cmd, Display  ↩
        display) {
        // Handle command parsing, execution or transmission of the cmd to a  ↩
            target system
        // here. The method should return immediately. A command response is  ↩
            optional.
        // If a CommandResponse is used it can be updated after this method  ↩
            returned to
        // indicate status of long running commands.

        // We just display the command string in a dialog for demonstration.
        JOptionPane.showMessageDialog(null, "ExampleCommandHandler. ↩
            issueStringTelecommand:"
            + cmd.getCanonicalString());

        // We return a CommandResponse with status "ok" to indicate successful  ↩
            execution.
        CommandResponse response = new CommandResponse(cmd);
        response.setStatus("OK");
        return response;
    }

    /**
     * Handle execution of a StructuredTeleCommand. The given cmd contains the  ↩
         command
     * string that is to be executed and parameters.
     */
    public CommandResponse issueStructuredTeleCommand(StructuredTeleCommand cmd,  ↩
        Display display) {
        // Handle execution or transmission of the cmd to a target system
        // here. The method should return immediately. A command response is  ↩
            optional.
        // If a CommandResponse is used it can be updated after this method  ↩
            returned to
        // indicate status of long running commands.

        // We just display the command string in a dialog for demonstration.
        JOptionPane.showMessageDialog(null, "ExampleCommandHandler. ↩
```

```
            issueStructuredTelecommand:"
            + cmd.getCanonicalString());

        // We return a CommandResponse with status "ok" to indicate successful ↩
            execution.
        CommandResponse response = new CommandResponse(cmd);
        response.setStatus("OK");
        return response;
    }

    /**
     * Handle confirmation of command execution. This method is called if an ↩
         interactive
     * command element is triggered be a user that has been flagged as guarded.
     */
    public boolean confirmCommandExecution(String title, String message,
        TeleCommand teleCommand, String targetSystem) {
        // This method has a default implementation in the executor.
        return true;
    }


    /**
     * Create a confirmation message for simple commands without parameters.
     */
    public String getConfirmationText(TeleCommand teleCommand, String ↩
        targetSystem) {
        // Return of a formatted confirmation message is optional.
        return null;
    }

    /**
     * Create a confirmation message for complex commands with parameters.
     */
    public String getEnhancedConfirmationText(TeleCommand teleCommand,
        String targetSystem, List<SimpleTeleCommandParameter> assocParameters) {
        // Return of a formatted confirmation message is optional.
        return null;
    }
}
```

The following table shows on which events these methods are called by the uss-executor.

Table B.3: TeleCommandHandler Methods and their Call Events

Method	**Event**
issueStringTeleCommand	called by the display with the command string to execute if an interactive command element with a string command has been triggered. String commands are usually used in a display for commands without interactive parameters.
issueStructuredTeleCommand	called by the display with the structured command to execute if an interactive command element with a string command has been triggered. Structured commands are usually used in a display for command with interactive parameters.
confirmCommandExecution	Not used by the executor. The executor has a built-in confirmation dialog.

Table B.3: (continued)

Method	Event
getConfirmationText	called when a guarded command element with a StringTeleCommand has been triggered on a display. This method can be used to assemble a confirmation message based on the TeleCommand contents.
getEnhancedConfirmationText	called when a guarded command element with a StructuredTeleCommand has been triggered on a display. This method can be used to assemble a confirmation message based on the TeleCommand contents.

Appendix C

Frequently Asked Questions (FAQ)

This appendix collects frequenty asked questions.

1. *What color names can I use in computations?*

   Allowed color names are listed in Section 9.4.

2. *I have defined my own colors in the editor's color picker. Can I use these names in computations?*

   No. The names in the color picker are for convenience only. They are not stored inside the display. The only allowed color names for computations are listed in Section 9.4.

3. *The quick line graph only shows 10 minutes of data. How can I see two hours of data?*

   In the executor, select Options → Preferences. In the Quick Graph page specify 7200 seconds (2 hours = 7200 seconds) and 0 samples (no limit on samples).

Appendix D

Glossary

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A

Active Symbol Symbol changing state or color reacting on an external stimuli (See IDAGS)

AD Applicable Document

ADP Acceptance Data Package

AIR Accident/Incident Report

Animated Symbol symbol continuously, dynamically changing its appearance without external stimuli (See IDAGS)

AP Agile Programming

AP Automated Procedure

API Application Programming Interface

APID Application Process Identifier.

APM Attached Pressurized Module

AR Acceptance Review (Formal acceptance of the whole system or parts of it)

ASCII American Standard Code for Information Interchange

B

BDUF Big Design Up Front

C

CC Control Center

CDR Critical Design Review (Design Freeze, Not foreseen in AP)

CGS Columbus Ground Software

COA Certificate of Acceptance

COC Certificate of Compliance

COL COLUMBUS Laboratory

COL-CC Columbus Control Centre

Command An order to initiate a change via the transmission of data (See IDAGS)

Command Button control initiating a TC command to be executed

Command Button Displays consist of different items. One item can be a command. A command is displayed as a round button. A guarded command as a round button marked with diagonal stripes. A display with lots of commands for example is the **COL Activation Part 1** display under `/examples/uss/fwdu/APM/FLTSYS/OPS/SYNOPTICS/ACTIVATION/ACT_PT_1.uss`.

Command List Displays consist of different items. One item can be a command list. A sample command list can be found in the home display COL Synoptics Home Page (under APM\FLTSYS\OPS\SYNOPTICS_ROO by clicking on the Set Cabin Temp square button.

Compound Symbol Any collection of primitives which are used together to denote a component, system, or function. Compound symbols are a special subset of icons that can be edited. (See IDAGS)

Computation A computation is a data source for elements such as data fields in a display. The value to be shown is determined by evaluating the computation's expression.

Control Display elements that are designed to cause a result when selected: command buttons, navigation buttons, and input controls. (See IDAGS)

CPL Common Procedure Language (Electronic test procedures with manual and automatic steps)

D

DaSS Data Service Sub-system

DDF Design Definition File (Design)

Decoration Standard features of a window to allow closing, resizing, iconification, and movement. (See IDAGS)

Default size During authoring displays are giving a default size. That is their normal, original size at which they look the best. So resetting the size of a display back to default size displays it again 1:1 without any zoom.

Display grouped set of data and information contained within a window (See IDAGS)

Display Element group of pixels assembled together to form an object, e.g., labels, symbols (See IDAGS)

DJF Design Justification File (Verification and Validation Plan, Test data)

DMS Data Management System

DN Discrepancy Note

DocBook DocBook provides a system for writing structured documents using XML. It is particularly well-suited to books and papers about computer software, though it is by no means limited to them. (www.docbook.org)

Data Quality Indicator (`DQI`)

A Data Quality Indicator is a letter or small symbol attached to a data field or icon to show the quality of a data value.

DQM Data Quality Monitor

DRD Document Requirement Description

E

EAC ESA Astronaut Centre

ECSS European Cooperation for Space Standardization (www.ecss.nl)

Enditem Box See Parameter Text Box

Engineering Task Can be everything e.g. part of a use case, user story, write chapter in a document, write test cases, develop a model. Task in iteration plan.

Expression An expression is part of a computation and defines the mathematical formula for evaluation.

F

FDB Flight Data Base

FE Flight Equipment

FEE Front End Equipment

FEECP Front End Equipment Communication Protocol

FMECA Failure Mode Effect and Criticality Analysis

FRC Facility Responsible Centre

FSC Facility Support Centre

FTP File Transfer Protocol

FWDU Flight Window Definition Utility (See CGS)

G

Graph Graphic image that shows the functional relationship of some quantities (See IDAGS)

Graphic Symbol denote the function, structure, and operation of different systems and components. Includes icons, compounds symbols, and primitives (See IDAGS)

Graphics same as graphic symbol (See IDAGS)

GSE Ground Support Equipment

GSS Ground Segment Simulator

GS-SRD Ground Segment System Requirement Document

GUI Graphical User Interface

GWDU Ground Window Definition Utility (See CGS)

H

HCI Human Computer Interface

Home Page provides a starting point and top-level information to begin operations and provides access to other displays (See IDAGS)

HSIA Hardware Software Interaction Analysis

I

I/F Interface

ICD Interface Control Document

Icon A small graphic symbol that uses a simplified picture to denote a system, component, state, or function. Examples: pumps, filters, and entire systems. (See IDAGS)

IDAGS International Space Station Display and Graphics Standards

IGS Interconnecting Ground System

ISS International Space Station

Iteration Plan Defines all engineering tasks for a specific iteration with their associated developer, initial estimate in ideal and real hours, real estimation to complete and priority within this iteration.

J

Java Programming language in the C/C++ family

K

KIP Key Inspection Point

L

Layout Layout means everything that changes the appearance of the executor and its loaded displays. Which are the window size, window position, open and loaded displays, status of tabbed mode, tab position, show hierarchy etc.

M

MCS Monitor and Control System (sub-system COL-CC)

MDB Mission Data Base

Menu a list of choices normally presented in a graphic form (See IDAGS)

MIN Minutes

MIP Mandatory Inspection Point

MMI Man Machine Interface

Mode Used to denote the current operational state of a system, subsystem, or device. (See IDAGS)

MRB Material Review Board

MTL Master Time Line

N

N/A Not Applicable

Navigation act of moving between displays (See IDAGS)

Navigation Button control for navigating to another display

NCR Non Conformance Report

Nickname Opsname and nickname are used as synonym

O

ODB Onboard Data Base

OPM Operations Manual

Opsname Opsname and nickname are used as synonym

P

P/L Payload

PA Product Assurance

PA/S Product Assurance and Safety

PAP Product Assurance Plan

Parameter Text Box A Display element for parameter output. Includes label, value, unit, quality indicator (See SRS)

PCE Proximity Communication Equipment

PCS Portable Computer System (NASA terminology)

PDR Preliminary Design Review (Review of the system architecture and requirements freeze)

PDF Invented by Adobe, Portable Document Format (PDF) is the published specification used around the world for more secure and reliable electronic document distribution and exchange. (www.adobe.com)

PFM Proto Flight Module

Plot Graphical representation of Data usually with a curve connecting data points and axes defining domain and range of the data.

PMP Parts Materials and Processes

PNG Portable Network Graphics

PREP PCS Reconfiguration Evolution Project (NASA terminology)

Primitive The simplest form of graphic available, e.g., circles, rectangles, lines, and pipes (See IDAGS)

Q

QA Quality Assurance

Quick Pick List A pull down menu used for sending commands that allows the user to select the desired value from a predefined list instead of entering it manually. (See IDAGS)

R

RAM Reliability, Availability, Maintainability

RB Requirements Baseline (See User Requirements Document)

RDB Result Data Base (archive of raw and processed data, and the event log)

Release Plan Also called commitment schedule, cycle plan (Highsmith99) or backlog (SCRUM). Defines the overall development release milestones. Assigned are the use cases which shall be developed for the specific milestones. Will be maintained over the time. Can also include use cases which are not scheduled for a release. Effort estimates are very rough in developer weeks. (See Development Plan)

RID Review Item Disposition

S

S/S Subsystem

S/W Software

SA Safety Analysis

SAS Special Application Software (interface via CGS API to CGS kernel)

SCA Software Criticality Analysis

SEEA Software Error Effect Analysis

SID Short Identifier

SMD Software Maintenance Disposition

SOW Statement of Work

SPA Software Problem Analysis

SPR System Problem Report / Software Problem Report

SRR System Requirements Review (See Review of user requirements and overall development planning)

SSMB Space Station Manned Base

SSO Safety Significant Operation

Stale When used as a status character, parameter is in the data stream but connection with the data stream has been lost. (See IDAGS)

State The physical configuration (On/Off, Open/Close, etc.) at the subsystem level or below. (See IDAGS)

Status A qualitative assessment of the overall condition or health of the system at any level. (See IDAGS)

SW Software

SWPA Software Product Assurance

Symbol see graphic symbol (See IDAGS)

T

TBD To Be Determined / Defined / Done

TC Telecommand

TES Test Evaluation Software (software performing real-time data acquisition, calibration, monitoring, automatic and manual procedure execution, command build, command verification)

TEV Test Evaluation Software (software to evaluate archived raw and processed data)

TIFF Tagged Image File Format

TM Telemetry

Tooltip Small informational pop up window that appears when the cursor is placed over a display object. (See IDAGS)

TQVS Training, Qualification and Validation Subsystem

TS Technical Specification (See Spec)

U

UCL User Control Language

UCLc User Control Language compiler

UHB User Home Base

UI User Interface

UML Unified Modeling Language (www.omg.org)

UP Unified Process

UR Usability Review (Present the system to the end users can also be performed instead of CDR or partial AR)

URL prefix The URL prefix is the location (e.g. server) where the help files lie. It is needed because all help links in the display are relative.

USS Unified Synoptic System

W

Window A portion of a screen that includes the display and its decoration. (See IDAGS)

WWW World Wide Web

X

XBM X-Bitmap (XBM) is an image file format.

XML The Extensible Markup Language (XML) is a W3C-recommended general-purpose markup language for creating special-purpose markup languages. It is a simplified subset of SGML, capable of describing many different kinds of data.

XP eXtreme Programming (www.extremeprogramming.org)

Appendix E

References

E.1 Reference Documents

[CGSSUM] *CGS 6.2 User Manual*, CGS Software User Manual.

[DGCS] *Display and Graphics Commonality Standard*

[OpNom05] *International Space Station Program*, Operations Nomenclature.

[OpNomESA04] *SSP 50254 Annex B1- Columbus*, ESA Specific Operations Nomenclature.

E.2 Other References

[JFormula] *JFormula home and specification*, http://www.japisoft.com/formula/.

Index

A
attributes
Supported attributes, 242

C
Check
Checking acquisition state, 178
Close
Closing all displays, 192
Closing display, 192
Closing other displays, 192
Command Response
Showing/Hiding Command Responses, 188
Computation, 149
Definition, 288
Configure
Configuring data quality indicators, 215
Configuring location of SCOE files, 5
Configuring status display, 214
Configuring System Settings, 4
Configuring user settings, 175
Connect
Connecting to system to be monitored and controlled, 176
Connection
Adapter, 275
Copy
Copying command to clipboard, 197
Copying parameter name to clipboard, 197
Create
Creating display snapshot, 213

D
Data Quality Indication, 10
Disconnect
Disconnecting system to be monitored and controlled, 177
Display
Definition, 288
Display Version, 245
Dock
Docking windows, 193
DQI, 10

E
Expression, 151
Definition, 289

F
FAQ
Frequently Asked Questions, 285
Find
Finding displays with parameter references, 201
Finding text in display, 202

G
Get
Getting the executor version information, 212
GWDU
GWDU to USS Conversion, 230
Importing GWDU displays, 229

I
Install
Installing the Product, 3
Issue
Issuing telecommand via command button, 197
Issuing telecommand via command list, 200

K
Keys
Format of Entry Keys, 242

L
Load
Loading display from file system, 187
Loading window layout, 186
localize
Translation work-flow, 241

M
MCS
Editor Configuration, 271
Executor Configuration, 272
MCS Configuration, 271

N
Navigate
Navigating display hierarchy, 189
Navigating system configuration browser, 189
Navigating to home display, 191

P
PCS
Importing PCS/PREP displays, 227
Preparation
Preparations, 24
Print
Print preview, 212
Printing display, 213
PWS
Importing PWS/FWDU displays, 228

Q
Quick Graph, 204

R
Reference
Menu references for the executor, 243
Reload
Reloading display from file system, 188
Reset
Resetting display window to default size, 188
Resize
Resizing display window, 188

RGB Colors, 258

S
Satmon
Importing Satmon displays, 233
Save
Saving a copy of current display, 214
Saving window layout, 185
Schema
XML Schema, 246
Show
Showing display help, 211
Showing display properties, 196
Showing element properties, 195
Showing parameter values in a quick graph, 203
Showing tooltip for element, 194
skeleton
Generating skeletons with the Skeleton Generator, 241
Start
Starting the Executor, 6
Switch
Switching target for commands, 177
Symbol Libraries
Concepts, 10

T
Toggle
Toggling tabbed mode, 193

U
Undock
Undocking windows, 193

V
View
Showing/Hiding the Toolbar, 192